

logisticregression

October 2, 2019

```
[5]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

```
[6]: train = pd.read_csv('titanic_train.csv')
train.head()
```

```
[6]: 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
```

```
[3]: train.count()
```

```
[3]: PassengerId    891
Survived          891
Pclass            891
Name              891
Sex               891
Age              714
```

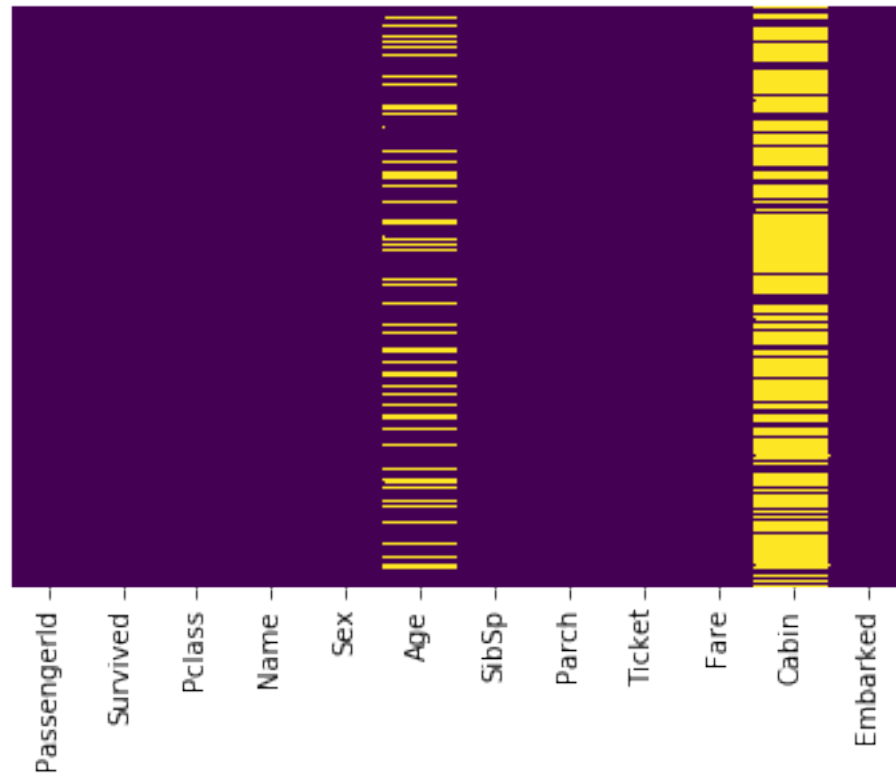
```
SibSp      891
Parch      891
Ticket     891
Fare       891
Cabin      204
Embarked   889
dtype: int64
```

```
[4]: train.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
```

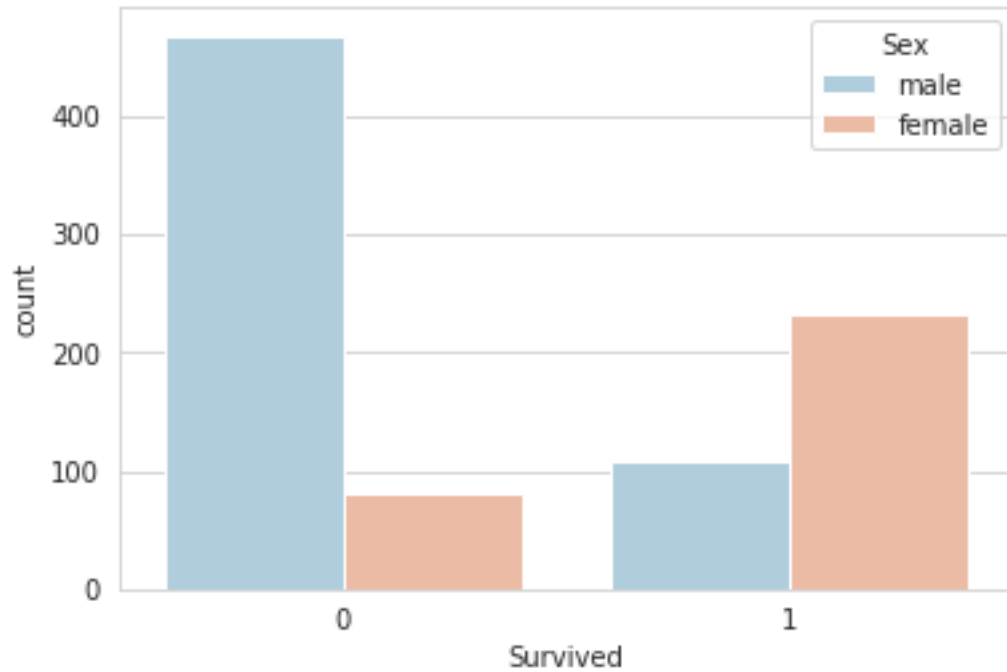
```
[7]: sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

```
[7]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0747f91ac8>
```



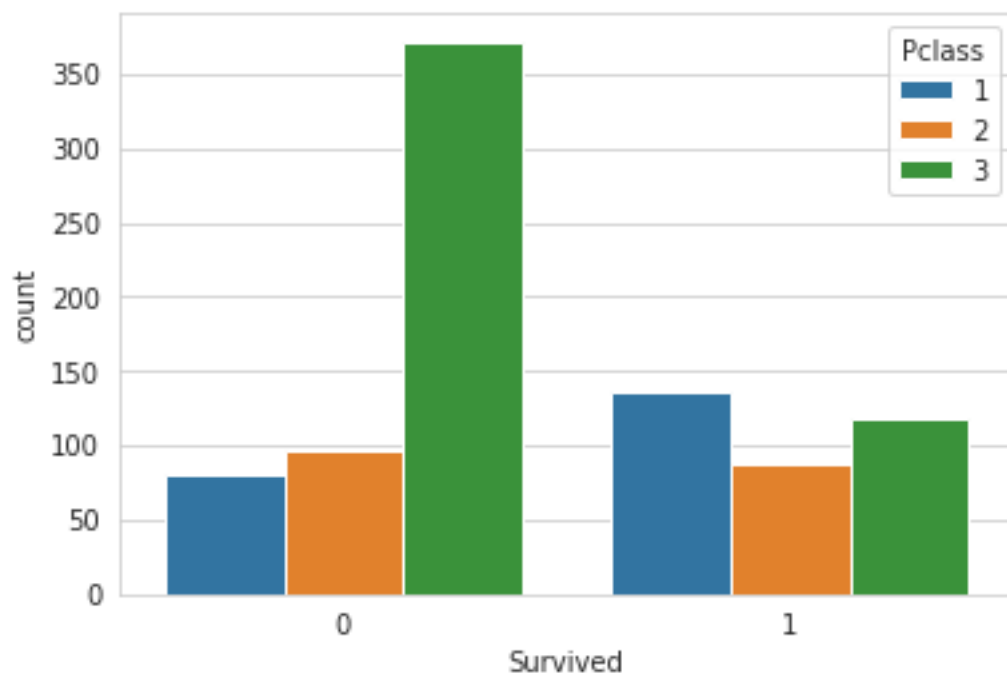
```
[9]: sns.set_style('whitegrid')
sns.countplot(x='Survived', hue='Sex', data=train, palette='RdBu_r')
```

```
[9]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0747ea2c18>
```



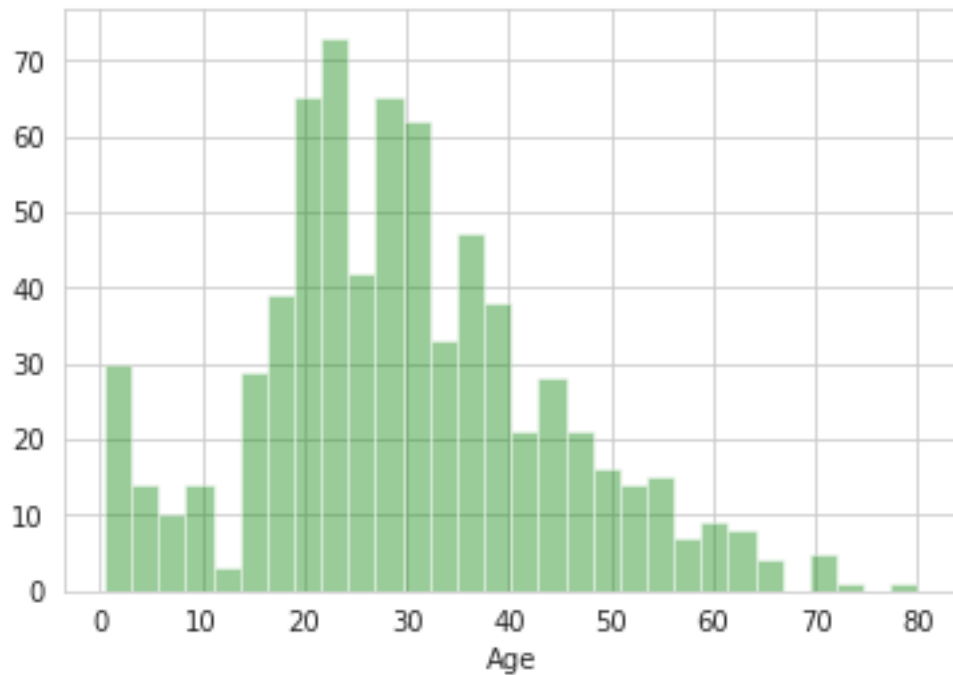
```
[10]: sns.set_style('whitegrid')  
sns.countplot(x='Survived', hue='Pclass', data=train)
```

```
[10]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0747bca390>
```



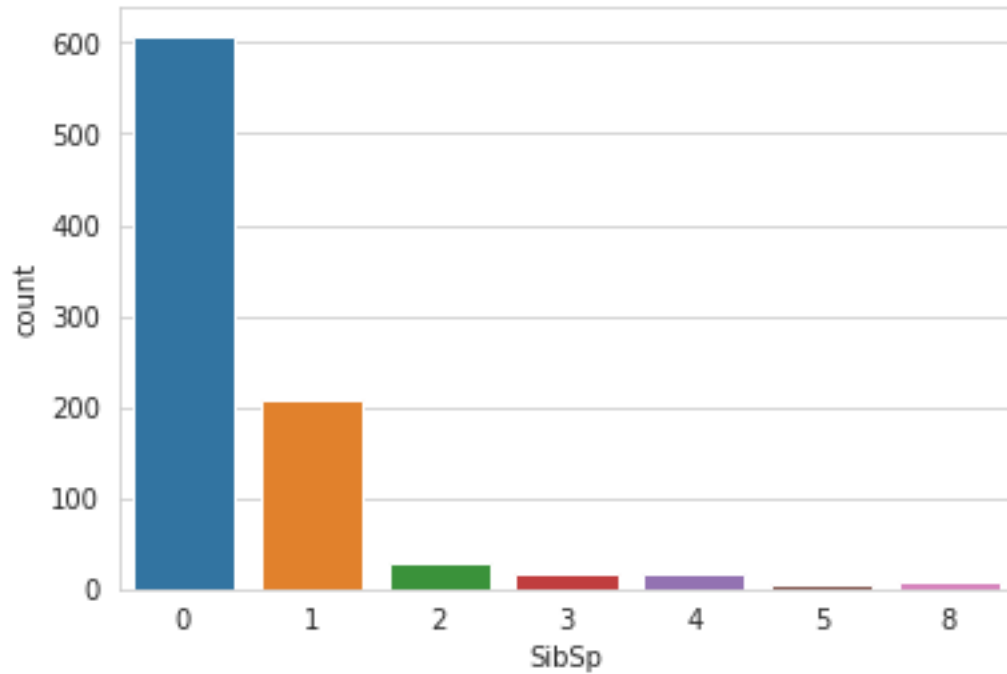
```
[11]: sns.distplot(train['Age'].dropna(), kde=False, bins=30, color='Green')
```

```
[11]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0747bbe240>
```



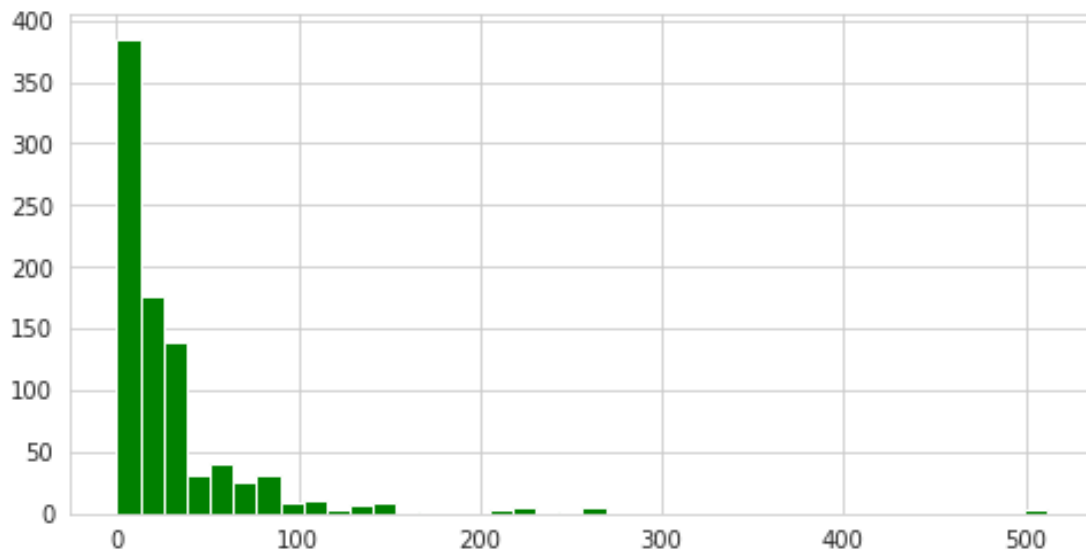
```
[12]: sns.countplot(x='SibSp', data=train)
```

```
[12]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0747ac3b00>
```



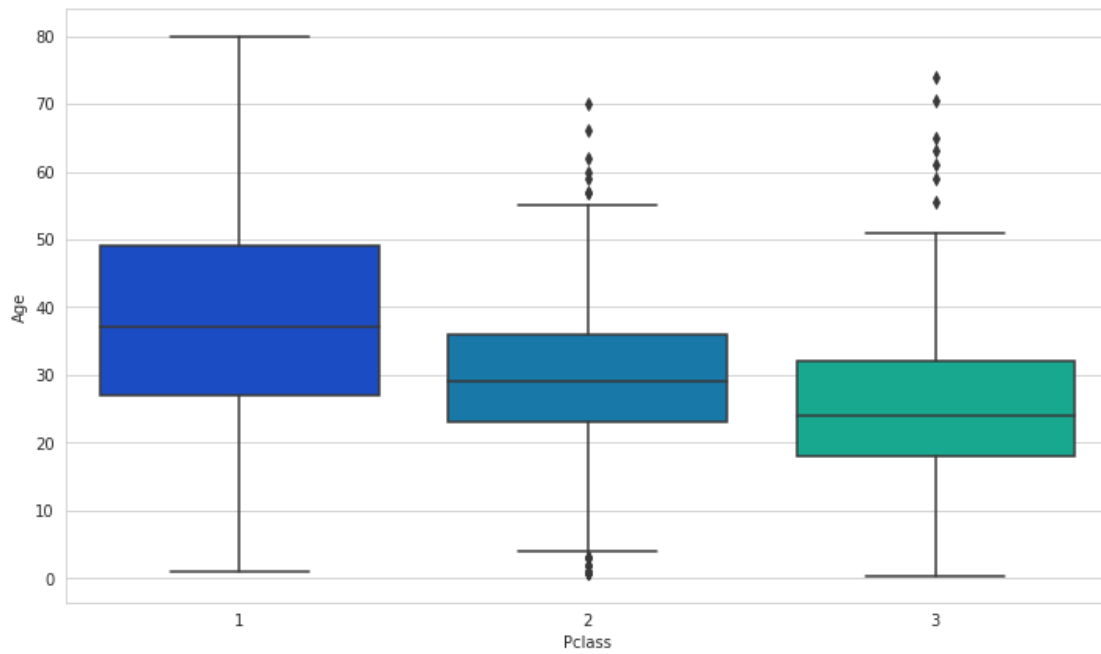
```
[13]: train['Fare'].hist(color='green',bins=40,figsize=(8,4))
```

```
[13]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0747b0ee80>
```



```
[14]: plt.figure(figsize=(12, 7))
sns.boxplot(x='Pclass',y='Age',data=train,palette='winter')
```

[14]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0747a4e9e8>



[17]: train=train.dropna()

[20]: train

[20]:

	PassengerId	Survived	Pclass	\
1	2	1	1	
3	4	1	1	
6	7	0	1	
10	11	1	3	
11	12	1	1	
21	22	1	2	
23	24	1	1	
27	28	0	1	
52	53	1	1	
54	55	0	1	
62	63	0	1	
66	67	1	2	
75	76	0	3	
88	89	1	1	
92	93	0	1	
96	97	0	1	
97	98	1	1	
102	103	0	1	
110	111	0	1	
118	119	0	1	

123	124	1	2
124	125	0	1
136	137	1	1
137	138	0	1
139	140	0	1
148	149	0	2
151	152	1	1
170	171	0	1
174	175	0	1
177	178	0	1
..
737	738	1	1
741	742	0	1
742	743	1	1
745	746	0	1
748	749	0	1
751	752	1	3
759	760	1	1
763	764	1	1
765	766	1	1
772	773	0	2
779	780	1	1
781	782	1	1
782	783	0	1
789	790	0	1
796	797	1	1
802	803	1	1
806	807	0	1
809	810	1	1
820	821	1	1
823	824	1	3
835	836	1	1
853	854	1	1
857	858	1	1
862	863	1	1
867	868	0	1
871	872	1	1
872	873	0	1
879	880	1	1
887	888	1	1
889	890	1	1

	Name	Sex	Age	SibSp	\
1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	
3	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	
6	McCarthy, Mr. Timothy J	male	54.0	0	
10	Sandstrom, Miss. Marguerite Rut	female	4.0	1	

11	Bonnell, Miss. Elizabeth	female	58.0	0
21	Beesley, Mr. Lawrence	male	34.0	0
23	Sloper, Mr. William Thompson	male	28.0	0
27	Fortune, Mr. Charles Alexander	male	19.0	3
52	Harper, Mrs. Henry Sleeper (Myna Haxtun)	female	49.0	1
54	Ostby, Mr. Engelhart Cornelius	male	65.0	0
62	Harris, Mr. Henry Birkhardt	male	45.0	1
66	Nye, Mrs. (Elizabeth Ramell)	female	29.0	0
75	Moen, Mr. Sigurd Hansen	male	25.0	0
88	Fortune, Miss. Mabel Helen	female	23.0	3
92	Chaffee, Mr. Herbert Fuller	male	46.0	1
96	Goldschmidt, Mr. George B	male	71.0	0
97	Greenfield, Mr. William Bertram	male	23.0	0
102	White, Mr. Richard Frasar	male	21.0	0
110	Porter, Mr. Walter Chamberlain	male	47.0	0
118	Baxter, Mr. Quigg Edmond	male	24.0	0
123	Webber, Miss. Susan	female	32.5	0
124	White, Mr. Percival Wayland	male	54.0	0
136	Newsom, Miss. Helen Monypeny	female	19.0	0
137	Futrelle, Mr. Jacques Heath	male	37.0	1
139	Giglio, Mr. Victor	male	24.0	0
148	Navratil, Mr. Michel ("Louis M Hoffman")	male	36.5	0
151	Pears, Mrs. Thomas (Edith Wearne)	female	22.0	1
170	Van der hoef, Mr. Wyckoff	male	61.0	0
174	Smith, Mr. James Clinch	male	56.0	0
177	Isham, Miss. Ann Elizabeth	female	50.0	0
..
737	Lesurer, Mr. Gustave J	male	35.0	0
741	Cavendish, Mr. Tyrell William	male	36.0	1
742	Ryerson, Miss. Susan Parker "Suzette"	female	21.0	2
745	Crosby, Capt. Edward Gifford	male	70.0	1
748	Marvin, Mr. Daniel Warner	male	19.0	1
751	Moor, Master. Meier	male	6.0	0
759	Roths, the Countess. of (Lucy Noel Martha Dye...	female	33.0	0
763	Carter, Mrs. William Ernest (Lucile Polk)	female	36.0	1
765	Hogeboom, Mrs. John C (Anna Andrews)	female	51.0	1
772	Mack, Mrs. (Mary)	female	57.0	0
779	Robert, Mrs. Edward Scott (Elisabeth Walton Mc...	female	43.0	0
781	Dick, Mrs. Albert Adrian (Vera Gillespie)	female	17.0	1
782	Long, Mr. Milton Clyde	male	29.0	0
789	Guggenheim, Mr. Benjamin	male	46.0	0
796	Leader, Dr. Alice (Farnham)	female	49.0	0
802	Carter, Master. William Thornton II	male	11.0	1
806	Andrews, Mr. Thomas Jr	male	39.0	0
809	Chambers, Mrs. Norman Campbell (Bertha Griggs)	female	33.0	1
820	Hays, Mrs. Charles Melville (Clara Jennings Gr...	female	52.0	1
823	Moor, Mrs. (Beila)	female	27.0	0

835		Compton, Miss. Sara Rebecca	female	39.0	1
853		Lines, Miss. Mary Conover	female	16.0	0
857		Daly, Mr. Peter Denis	male	51.0	0
862	Swift, Mrs. Frederick Joel (Margaret Welles Ba...		female	48.0	0
867		Roebling, Mr. Washington Augustus II	male	31.0	0
871	Beckwith, Mrs. Richard Leonard (Sallie Monypeny)		female	47.0	1
872		Carlsson, Mr. Frans Olof	male	33.0	0
879	Potter, Mrs. Thomas Jr (Lily Alexenia Wilson)		female	56.0	0
887		Graham, Miss. Margaret Edith	female	19.0	0
889		Behr, Mr. Karl Howell	male	26.0	0

	Parch	Ticket	Fare	Cabin	Embarked
1	0	PC 17599	71.2833	C85	C
3	0	113803	53.1000	C123	S
6	0	17463	51.8625	E46	S
10	1	PP 9549	16.7000	G6	S
11	0	113783	26.5500	C103	S
21	0	248698	13.0000	D56	S
23	0	113788	35.5000	A6	S
27	2	19950	263.0000	C23 C25 C27	S
52	0	PC 17572	76.7292	D33	C
54	1	113509	61.9792	B30	C
62	0	36973	83.4750	C83	S
66	0	C.A. 29395	10.5000	F33	S
75	0	348123	7.6500	F G73	S
88	2	19950	263.0000	C23 C25 C27	S
92	0	W.E.P. 5734	61.1750	E31	S
96	0	PC 17754	34.6542	A5	C
97	1	PC 17759	63.3583	D10 D12	C
102	1	35281	77.2875	D26	S
110	0	110465	52.0000	C110	S
118	1	PC 17558	247.5208	B58 B60	C
123	0	27267	13.0000	E101	S
124	1	35281	77.2875	D26	S
136	2	11752	26.2833	D47	S
137	0	113803	53.1000	C123	S
139	0	PC 17593	79.2000	B86	C
148	2	230080	26.0000	F2	S
151	0	113776	66.6000	C2	S
170	0	111240	33.5000	B19	S
174	0	17764	30.6958	A7	C
177	0	PC 17595	28.7125	C49	C
...
737	0	PC 17755	512.3292	B101	C
741	0	19877	78.8500	C46	S
742	2	PC 17608	262.3750	B57 B59 B63 B66	C
745	1	WE/P 5735	71.0000	B22	S

748	0	113773	53.1000	D30	S
751	1	392096	12.4750	E121	S
759	0	110152	86.5000	B77	S
763	2	113760	120.0000	B96 B98	S
765	0	13502	77.9583	D11	S
772	0	S.O./P.P. 3	10.5000	E77	S
779	1	24160	211.3375	B3	S
781	0	17474	57.0000	B20	S
782	0	113501	30.0000	D6	S
789	0	PC 17593	79.2000	B82 B84	C
796	0	17465	25.9292	D17	S
802	2	113760	120.0000	B96 B98	S
806	0	112050	0.0000	A36	S
809	0	113806	53.1000	E8	S
820	1	12749	93.5000	B69	S
823	1	392096	12.4750	E121	S
835	1	PC 17756	83.1583	E49	C
853	1	PC 17592	39.4000	D28	S
857	0	113055	26.5500	E17	S
862	0	17466	25.9292	D17	S
867	0	PC 17590	50.4958	A24	S
871	1	11751	52.5542	D35	S
872	0	695	5.0000	B51 B53 B55	S
879	1	11767	83.1583	C50	C
887	0	112053	30.0000	B42	S
889	0	111369	30.0000	C148	C

[183 rows x 12 columns]

```
[21]: train.drop('Cabin',axis=1,inplace=True)
train.dropna(inplace=True)
train.head()
```

```
[21]: PassengerId  Survived  Pclass  \
1             2         1         1
3             4         1         1
6             7         0         1
10            11         1         3
11            12         1         1
```

	Name	Sex	Age	SibSp	\
1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	
3	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	
6	McCarthy, Mr. Timothy J	male	54.0	0	
10	Sandstrom, Miss. Marguerite Rut	female	4.0	1	
11	Bonnell, Miss. Elizabeth	female	58.0	0	

Parch	Ticket	Fare	Embarked
-------	--------	------	----------

1	0	PC	17599	71.2833	C
3	0		113803	53.1000	S
6	0		17463	51.8625	S
10	1	PP	9549	16.7000	S
11	0		113783	26.5500	S

```
[22]: sex = pd.get_dummies(train['Sex'],drop_first=True)
embark = pd.get_dummies(train['Embarked'],drop_first=True)
#drop the sex,embarked,name and tickets columns
train.drop(['Sex','Embarked','Name','Ticket'],axis=1,inplace=True)
#concatenate new sex and embark column to our train dataframe
train = pd.concat([train,sex,embark],axis=1)
#check the head of dataframe
train.head()
```

```
[22]: PassengerId  Survived  Pclass   Age  SibSp  Parch    Fare   male  Q  S
1             2         1       1  38.0     1     0  71.2833    0  0  0
3             4         1       1  35.0     1     0  53.1000    0  0  1
6             7         0       1  54.0     0     0  51.8625    1  0  1
10            11         1       3   4.0     1     1  16.7000    0  0  1
11            12         1       1  58.0     0     0  26.5500    0  0  1
```

```
[25]: from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(train.
→drop('Survived',axis=1), train['Survived'], test_size=0.30,random_state=101)
```

```
[26]: from sklearn.linear_model import LogisticRegression
#create an instance and fit the model
logmodel = LogisticRegression()
logmodel.fit(X_train, y_train)
```

```
/home/nbuser/anaconda3_501/lib/python3.6/site-
packages/sklearn/linear_model/logistic.py:433: FutureWarning: Default solver
will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
FutureWarning)
```

```
[26]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
intercept_scaling=1, max_iter=100, multi_class='warn',
n_jobs=None, penalty='l2', random_state=None, solver='warn',
tol=0.0001, verbose=0, warm_start=False)
```

```
[27]: Predictions = logmodel.predict(X_test)
```

```
[29]: from sklearn.metrics import classification_report
print(classification_report(y_test,Predictions))
```

	precision	recall	f1-score	support
0	0.62	0.45	0.53	22
1	0.69	0.82	0.75	33

micro avg	0.67	0.67	0.67	55
macro avg	0.66	0.64	0.64	55
weighted avg	0.67	0.67	0.66	55

```
[32]: from sklearn.metrics import confusion_matrix
print(confusion_matrix(y_test, Predictions))
```

```
[[10 12]
 [ 6 27]]
```

```
[33]: train
```

```
[33]: PassengerId  Survived  Pclass   Age  SibSp  Parch    Fare   male    Q    S
1             2         1       1  38.0     1     0   71.2833    0    0    0
3             4         1       1  35.0     1     0   53.1000    0    0    1
6             7         0       1  54.0     0     0   51.8625    1    0    1
10            11         1       3   4.0     1     1   16.7000    0    0    1
11            12         1       1  58.0     0     0   26.5500    0    0    1
21            22         1       2  34.0     0     0   13.0000    1    0    1
23            24         1       1  28.0     0     0   35.5000    1    0    1
27            28         0       1  19.0     3     2  263.0000    1    0    1
52            53         1       1  49.0     1     0   76.7292    0    0    0
54            55         0       1  65.0     0     1   61.9792    1    0    0
62            63         0       1  45.0     1     0   83.4750    1    0    1
66            67         1       2  29.0     0     0   10.5000    0    0    1
75            76         0       3  25.0     0     0    7.6500    1    0    1
88            89         1       1  23.0     3     2  263.0000    0    0    1
92            93         0       1  46.0     1     0   61.1750    1    0    1
96            97         0       1  71.0     0     0   34.6542    1    0    0
97            98         1       1  23.0     0     1   63.3583    1    0    0
102           103         0       1  21.0     0     1   77.2875    1    0    1
110           111         0       1  47.0     0     0   52.0000    1    0    1
118           119         0       1  24.0     0     1  247.5208    1    0    0
123           124         1       2  32.5     0     0   13.0000    0    0    1
124           125         0       1  54.0     0     1   77.2875    1    0    1
136           137         1       1  19.0     0     2   26.2833    0    0    1
137           138         0       1  37.0     1     0   53.1000    1    0    1
139           140         0       1  24.0     0     0   79.2000    1    0    0
148           149         0       2  36.5     0     2   26.0000    1    0    1
151           152         1       1  22.0     1     0   66.6000    0    0    1
170           171         0       1  61.0     0     0   33.5000    1    0    1
174           175         0       1  56.0     0     0   30.6958    1    0    0
177           178         0       1  50.0     0     0   28.7125    0    0    0
..           ...         ...     ...     ...     ...     ...     ...     ... ..
737           738         1       1  35.0     0     0  512.3292    1    0    0
741           742         0       1  36.0     1     0   78.8500    1    0    1
```

742	743	1	1	21.0	2	2	262.3750	0	0	0
745	746	0	1	70.0	1	1	71.0000	1	0	1
748	749	0	1	19.0	1	0	53.1000	1	0	1
751	752	1	3	6.0	0	1	12.4750	1	0	1
759	760	1	1	33.0	0	0	86.5000	0	0	1
763	764	1	1	36.0	1	2	120.0000	0	0	1
765	766	1	1	51.0	1	0	77.9583	0	0	1
772	773	0	2	57.0	0	0	10.5000	0	0	1
779	780	1	1	43.0	0	1	211.3375	0	0	1
781	782	1	1	17.0	1	0	57.0000	0	0	1
782	783	0	1	29.0	0	0	30.0000	1	0	1
789	790	0	1	46.0	0	0	79.2000	1	0	0
796	797	1	1	49.0	0	0	25.9292	0	0	1
802	803	1	1	11.0	1	2	120.0000	1	0	1
806	807	0	1	39.0	0	0	0.0000	1	0	1
809	810	1	1	33.0	1	0	53.1000	0	0	1
820	821	1	1	52.0	1	1	93.5000	0	0	1
823	824	1	3	27.0	0	1	12.4750	0	0	1
835	836	1	1	39.0	1	1	83.1583	0	0	0
853	854	1	1	16.0	0	1	39.4000	0	0	1
857	858	1	1	51.0	0	0	26.5500	1	0	1
862	863	1	1	48.0	0	0	25.9292	0	0	1
867	868	0	1	31.0	0	0	50.4958	1	0	1
871	872	1	1	47.0	1	1	52.5542	0	0	1
872	873	0	1	33.0	0	0	5.0000	1	0	1
879	880	1	1	56.0	0	1	83.1583	0	0	0
887	888	1	1	19.0	0	0	30.0000	0	0	1
889	890	1	1	26.0	0	0	30.0000	1	0	0

[183 rows x 10 columns]

1 KMEANS

```
[34]: X = np.array(train.drop(['Survived'], 1).astype(float))
```

```
[35]: y = np.array(train['Survived'])
```

```
[37]: from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=2)
kmeans.fit(X)
```

```
[37]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
n_clusters=2, n_init=10, n_jobs=None, precompute_distances='auto',
random_state=None, tol=0.0001, verbose=0)
```

```
[38]: correct = 0
for i in range(len(X)):
    predict_me = np.array(X[i].astype(float))
```

```
predict_me = predict_me.reshape(-1, len(predict_me))
prediction = kmeans.predict(predict_me)
if prediction[0] == y[i]:
    correct += 1

print(correct/len(X))
```

0.4426229508196721

```
[42]: from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)
```

```
[43]: kmeans.fit(X_scaled)
```

```
[43]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
n_clusters=2, n_init=10, n_jobs=None, precompute_distances='auto',
random_state=None, tol=0.0001, verbose=0)
```

```
[44]: correct = 0
for i in range(len(X)):
    predict_me = np.array(X[i].astype(float))
    predict_me = predict_me.reshape(-1, len(predict_me))
    prediction = kmeans.predict(predict_me)
    if prediction[0] == y[i]:
        correct += 1

print(correct/len(X))
```

0.6994535519125683

```
[ ]:
```