CHANDAN KUMAR

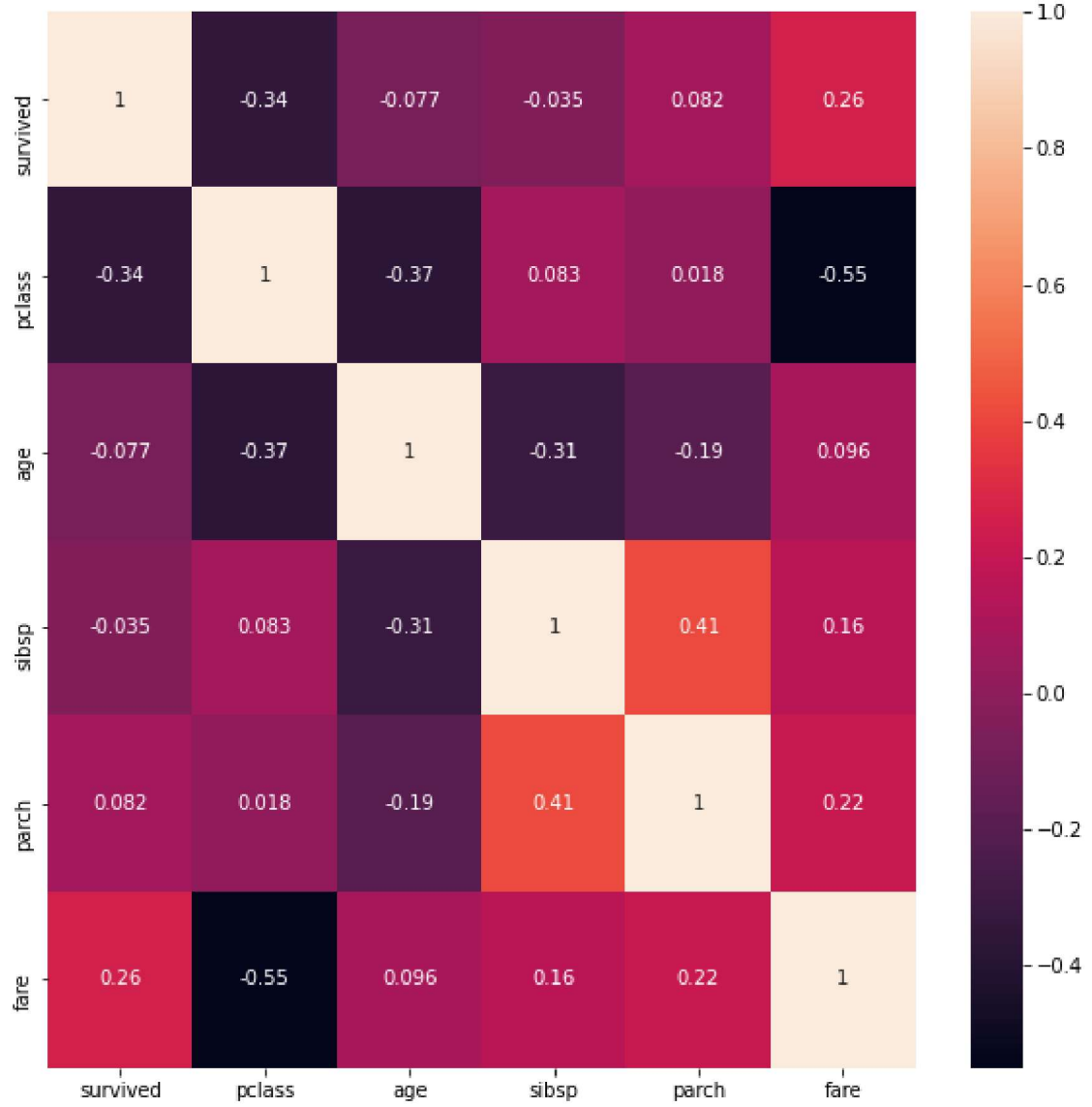
ID: GO_STP_13267

```
1 import numpy as np
2 from sklearn.model_selection import train_test_split
3 import matplotlib.pyplot as plt
4 import pandas as pd
5 import seaborn as sns
```

```
1 url = "https://raw.github.com/mattdelhey/kaggle-titanic/master/Data/train.csv"
2 df = pd.read_csv(url)
```

```
1 plt.figure(figsize=(10, 10))
2 sns.heatmap(df.corr(), annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdcebe29750>
```



```
df.isna().sum()
```

```
survived      0
pclass        0
name          0
sex           0
age         177
sibsp         0
parch         0
ticket        0
fare          0
cabin       687
embarked      2
dtype: int64
```

```
df.isnull().sum()
```

```
survived      0
pclass        0
name          0
sex           0
age         177
sibsp         0
parch         0
ticket        0
fare          0
cabin       687
embarked      2
dtype: int64
```

```
def fun(sex):
    return dict(male=1, female=0)[sex]


df['sex'] = df['sex'].apply(fun)
```

```
df.dropna(axis=0, inplace=True)
```

```
def emba(feature):
    return dict(S=3, Q=2, C=1)[feature]


df['embarked'] = df['embarked'].apply(emba)
```

```
df['age'] = df['age'].fillna(np.round(df['age'].mean(), 0))
```

```
df.drop(labels='cabin', axis=1, inplace=True)
```

```
df['fare'] = np.round(df['fare'], 2)
df.drop(labels='ticket', axis=1, inplace=True)
df.drop(labels='name', axis=1, inplace=True)
```

```
y = df['survived']
X = df.drop(labels='survived', axis=1)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

```
from sklearn.metrics import accuracy_score, confusion_matrix, f1_score, recall_score, p
```

```
from sklearn.linear_model import LogisticRegression
model_1 = LogisticRegression()
model_1.fit(X_train, y_train)
m, n = model_1.predict(X_test), y_test
print(confusion_matrix(m, n))
print("Accuracy: ", accuracy_score(m, n) * 100, "%")
print("Precision: ", precision_score(m, n) * 100, "%")
print("Recall: ", recall_score(m, n))
print("F1: ", f1_score(m, n))
```

```
[[10  7]
 [11 27]]
Accuracy:  67.27272727272727 %
Precision:  79.41176470588235 %
Recall:  0.7105263157894737
F1:  0.7499999999999999
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: Converg
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

```
# final model
from sklearn.ensemble import RandomForestClassifier
model_2 = RandomForestClassifier(random_state=547)
model_2.fit(X_train, y_train)
m, n = model_2.predict(X_test), y_test
print(confusion_matrix(m, n))
acc = accuracy_score(m, n)
pre = precision_score(m, n)
print("Accuracy: ", acc * 100, "%")
print("Precision: ", pre * 100, "%")
print("Recall: ", recall_score(m, n))
print("F1: ", f1_score(m, n))
```

```
[[10  6]
 [11 28]]
Accuracy:  69.0909090909091 %
Precision:  82.35294117647058 %
Recall:  0.717948717948718
F1:  0.767123287671233
```

```
from sklearn.tree import DecisionTreeClassifier
```

```
model_3 = DecisionTreeClassifier(random_state=547)
model_3.fit(X_train, y_train)
m, n = model_3.predict(X_test), y_test
print(confusion_matrix(m, n))
acc = accuracy_score(m, n)
pre = precision_score(m, n)
print("Accuracy: ", acc * 100, "%")
print("Precision: ", pre * 100, "%")
print("Recall: ", recall_score(m, n))
print("F1: ", f1_score(m, n))
```

```
[[15  9]
 [ 6 25]]
Accuracy:  72.72727272727273 %
Precision:  73.52941176470588 %
Recall:  0.8064516129032258
F1:  0.7692307692307693
```

✓ 0s completed at 6:06 PM