2. Titanic

March 31, 2022

[1]:
```
# Stages of Data Cleaning
# 1. Parse date/time (if time-stamps present)
# 2. Drop un-necessary columns (for machine learning) like ID, etc.
# 3. Feature Engineering (derive new features from existing features)
# 4. Data imputation: Fill the missing values (with mode, median, mean, etc.)␣
 ↪OR Drop rows with missing values
# 3. Handling class imbalance: oversampling e.g. SMOTE, ADASYN
# 5. One Hot Encoding of categorical features
# 6. Normalization (0-1) / Standarization (mean=0, SD=1)
# 7. PDimensionality Reduction: Feature Transformation (PCA/t-SNE) or Feature␣
 ↪selection (chiq-square test, RFE, etc. )
# After data cleaning we can perform:
# Data Modeling (Machine Learning + Regularization) with Hyper-parameter Tuning␣
 ↪(Grid search)
# Model Evaluation (Accuracy, Precision, F1 score, Confusion Matrix, AUC) and␣
 ↪Visualization
```

[2]:
```
import pandas as pd
import numpy as np
train_df = pd.read_csv("titanic_train.csv")
test_df = pd.read_csv("titanic_test.csv")
train_df.head(2).transpose()
```

[2]:

	0
PassengerId	1
Survived	0
Pclass	3
Name	Braund, Mr. Owen Harris
Sex	male
Age	22.0
SibSp	1
Parch	0
Ticket	A/5 21171
Fare	7.25
Cabin	NaN
Embarked	S

```
                                                                1
PassengerId                                                     2
Survived                                                        1
Pclass                                                          1
Name         Cumings, Mrs. John Bradley (Florence Briggs Th…
Sex                                                        female
Age                                                          38.0
SibSp                                                           1
Parch                                                           0
Ticket                                                   PC 17599
Fare                                                      71.2833
Cabin                                                         C85
Embarked                                                      C
```

[3]:
```
# check for missing values in all the columns
train_df.isnull().sum()
```

[3]:
```
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
```

[4]:
```
test_df.isnull().sum()
```

[4]:
```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

[5]:
```
embarked_mode = train_df['Embarked'].mode()
data = [train_df, test_df]
for dataset in data:
    dataset['Embarked'] = dataset['Embarked'].fillna(embarked_mode)
```

[6]:
```
data = [train_df, test_df]
for dataset in data:
    mean = train_df["Age"].mean()
    std = test_df["Age"].std()
    is_null = dataset["Age"].isnull().sum()
    # compute random numbers between the mean, std and is_null
    rand_age = np.random.randint(mean - std, mean + std, size = is_null)
    # fill NaN values in Age column with random values generated
    age_slice = dataset["Age"].copy()
    age_slice[np.isnan(age_slice)] = rand_age
    dataset["Age"] = age_slice
    dataset["Age"] = train_df["Age"].astype(int)
```

[7]:
```
data = [train_df, test_df]
for dataset in data:
    dataset['relatives'] = dataset['SibSp'] + dataset['Parch']
    dataset.loc[dataset['relatives'] > 0, 'travelled_alone'] = 'No'
    dataset.loc[dataset['relatives'] == 0, 'travelled_alone'] = 'Yes'
```

[8]:
```
for dataset in data:
    dataset.loc[ dataset['Age'] <= 16, 'Age'] = 0
    dataset.loc[(dataset['Age'] > 16) & (dataset['Age'] <= 32), 'Age'] = 1
    dataset.loc[(dataset['Age'] > 32) & (dataset['Age'] <= 48), 'Age'] = 2
    dataset.loc[(dataset['Age'] > 48) & (dataset['Age'] <= 64), 'Age'] = 3
    dataset.loc[ dataset['Age'] > 64, 'Age']
train_df.head(2).transpose()
```

[8]:

	0
PassengerId	1
Survived	0
Pclass	3
Name	Braund, Mr. Owen Harris
Sex	male
Age	1
SibSp	1
Parch	0
Ticket	A/5 21171
Fare	7.25
Cabin	NaN
Embarked	S
relatives	1
travelled_alone	No

```
                                                                  1
PassengerId                                                       2
Survived                                                          1
Pclass                                                            1
Name             Cumings, Mrs. John Bradley (Florence Briggs Th…
Sex                                                          female
Age                                                               2
SibSp                                                             1
Parch                                                             0
Ticket                                                    PC 17599
Fare                                                       71.2833
Cabin                                                          C85
Embarked                                                         C
relatives                                                        1
travelled_alone                                                 No
```

[9]:
```
train_df = train_df.
↪drop(['PassengerId','Name','Ticket','Cabin','SibSp','Parch'], axis=1)
train_df.head()
```

[9]:

	Survived	Pclass	Sex	Age	Fare	Embarked	relatives	travelled_alone
0	0	3	male	1	7.2500	S	1	No
1	1	1	female	2	71.2833	C	1	No
2	1	3	female	1	7.9250	S	0	Yes
3	1	1	female	2	53.1000	S	1	No
4	0	3	male	2	8.0500	S	0	Yes

[10]:
```
test_df = test_df.drop(['Name','Ticket','Cabin','SibSp','Parch'], axis=1)
test_df.head()
```

[10]:

	PassengerId	Pclass	Sex	Age	Fare	Embarked	relatives	travelled_alone
0	892	3	male	1	7.8292	Q	0	Yes
1	893	3	female	2	7.0000	S	1	No
2	894	2	male	1	9.6875	Q	0	Yes
3	895	3	male	2	8.6625	S	0	Yes
4	896	3	female	2	12.2875	S	2	No

[11]:
```
numerical_features = list(train_df.select_dtypes(include=['int64', 'float64',␣
↪'int32']).columns)
```

```
numerical_features
```

[11]:
```
['Survived', 'Pclass', 'Age', 'Fare', 'relatives']
```

[12]:
```
del numerical_features[0]
numerical_features
```

[12]:
```
['Pclass', 'Age', 'Fare', 'relatives']
```

[13]:
```
from sklearn.preprocessing import StandardScaler
ss_scaler = StandardScaler()
train_df_ss = pd.DataFrame(data = train_df)
train_df_ss[numerical_features] = ss_scaler.
 ↪fit_transform(train_df_ss[numerical_features])
train_df_ss.head()
```

[13]:

	Survived	Pclass	Sex	Age	Fare	Embarked	relatives	travelled_alone
0	0	0.827377	male	-0.149052	-0.502445	S	0.059160	No
1	1	-1.566107	female	-0.017692	0.786845	C	0.059160	No
2	1	0.827377	female	-0.149052	-0.488854	S	-0.560975	Yes
3	1	-1.566107	female	-0.017692	0.420730	S	0.059160	No
4	0	0.827377	male	-0.017692	-0.486337	S	-0.560975	Yes

[14]:
```
encode_col_list = list(train_df.select_dtypes(include=['object']).columns)
encode_col_list
```

[14]:
```
['Sex', 'Embarked', 'travelled_alone']
```

[15]:
```
for i in encode_col_list:
    train_df_ss = pd.concat([train_df_ss,pd.get_dummies(train_df_ss[i],␣
 ↪prefix=i)],axis=1)
    train_df_ss.drop(i, axis = 1, inplace=True)
train_df_ss.head(2).transpose()
```

[15]:

	0	1
Survived	0.000000	1.000000
Pclass	0.827377	-1.566107
Age	-0.149052	-0.017692
Fare	-0.502445	0.786845
relatives	0.059160	0.059160

```
Sex_female           0.000000  1.000000
Sex_male             1.000000  0.000000
Embarked_C           0.000000  1.000000
Embarked_Q           0.000000  0.000000
Embarked_S           1.000000  0.000000
travelled_alone_No   1.000000  1.000000
travelled_alone_Yes  0.000000  0.000000
```

[16]:
```
numerical_features = list(test_df.select_dtypes(include=['int64', 'float64',␣
 ↪'int32']).columns)
del numerical_features[0]
ss_scaler = StandardScaler()
test_df_ss = pd.DataFrame(data = test_df)
test_df_ss[numerical_features] = ss_scaler.
 ↪fit_transform(test_df_ss[numerical_features])
test_df_ss.head()
```

[16]:

	PassengerId	Pclass	Sex	Age	Fare	Embarked	relatives	travelled_alone
0	892	0.873482	male	-0.142784	-0.497811	Q	-0.553443	Yes
1	893	0.873482	female	-0.004627	-0.512660	S	0.105643	No
2	894	-0.315819	male	-0.142784	-0.464532	Q	-0.553443	Yes
3	895	0.873482	male	-0.004627	-0.482888	S	-0.553443	Yes
4	896	0.873482	female	-0.004627	-0.417971	S	0.764728	No

[17]:
```
test_encode_col_list = list(test_df.select_dtypes(include=['object']).columns)
for i in test_encode_col_list:
    test_df_ss = pd.concat([test_df_ss,pd.get_dummies(test_df_ss[i],␣
 ↪prefix=i)],axis=1)
    test_df_ss.drop(i, axis = 1, inplace=True)
test_df_ss.head(2).transpose()
```

[17]:

	0	1
PassengerId	892.000000	893.000000
Pclass	0.873482	0.873482
Age	-0.142784	-0.004627
Fare	-0.497811	-0.512660
relatives	-0.553443	0.105643
Sex_female	0.000000	1.000000
Sex_male	1.000000	0.000000
Embarked_C	0.000000	0.000000

```
Embarked_Q              1.000000    0.000000
Embarked_S              0.000000    1.000000
travelled_alone_No      0.000000    1.000000
travelled_alone_Yes     1.000000    0.000000
```

[18]:
```
X_train = train_df_ss.drop("Survived", axis=1)
Y_train = train_df_ss["Survived"]
X_test  = test_df_ss.drop("PassengerId", axis=1).copy()
```

[19]:
```
X_test = X_test.fillna(0)
#X_test[X_test.isin([np.nan, np.inf, -np.inf]).any(1)]
```

[20]:
```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()  # Fit our model to the training data
logreg.fit(X_train, Y_train)   # Predict on the test data
logreg_predictions = logreg.predict(X_test)
logreg_predictions
```

[20]:
```
array([0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0,
       1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1,
       1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1,
       1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1,
       1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
       0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1,
       1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1,
       1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
       0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0,
       1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0],
      dtype=int64)
```

[21]:
```
logreg_data = pd.read_csv('titanic_test.csv')
logreg_data.insert((logreg_data.shape[1]),'Survived',logreg_predictions)
logreg_data.head(2).transpose()
#logreg_data.to_csv('LogisticRegression_SS_OH_FE2.csv')
```

[21]:

	0	1
PassengerId	892	893
Pclass	3	3

Name	Kelly, Mr. James	Wilkes, Mrs. James (Ellen Needs)
Sex	male	female
Age	34.5	47.0
SibSp	0	1
Parch	0	0
Ticket	330911	363272
Fare	7.8292	7.0
Cabin	NaN	NaN
Embarked	Q	S
Survived	0	1

[ ]: