

DS6 ASSIGNMENT INTRODUCTION

Created by- Rajeshwar Sharma

Created On- 19/09/2020

Overview:

The project focus on predicting the survival rate for the titanic disaster. Here we are building a predictive model that answers the question: "what sorts of people were more likely to survive?" using passenger data (ie name, age, gender, socio-economic class, etc).

Source:

<https://www.kaggle.com/c/titanic/data>

Structure of the project:

This project structure contains 4 major directories that we mostly deal with.

1. *data* - it has train, test, raw & processed data
2. *notebook* - contains all the .ipyn jupyter notebooks
3. *src* - python code that is used to process data, make submissions, get the source data, perform feature engineering, and so on.
4. Apart from that, there are a couple of other files that are necessary such as *test_environment.py*, *requirement.txt*

Below is the screenshot for the complete folder structure for data science projects:

```
— README.md
— data
  — external
    — submission_data_01.csv
    — submission_data_02.csv
  — get_processed_data.py
  — interim
  — processed
    — test.csv
    — train.csv
  — raw
    — test.csv
    — titanic.zip
    — train.csv
— docs
  — Makefile
  — commands.rst
  — conf.py
  — getting-started.rst
  — index.rst
— models
— notebooks
  — extracting-data-using-python.ipynb
  — notebookpdf
    — extracting-data-using-python.pdf
    — predictive-model-using-python.pdf
    — processing-data-using-python-I.pdf
    — processing-data-using-python-II.pdf
  — predictive-model-using-python.ipynb
  — processing-data-using-python-1.0.ipynb
  — processing-data-using-python-1.1.ipynb
  — test.csv
  — train.csv
— references
— reports
  — figures
— requirements.txt
— setup.py
— src
  — __init__.py
  — data
    — __init__.py
    — get_processed_data.py
    — get_raw_data.py
    — make_dataset.py
  — features
    — __init__.py
    — build_features.py
  — models
    — __init__.py
    — predict_model.py
    — train_model.py
  — visualization
    — __init__.py
    — visualize.py
— test_environment.py
```

Notebooks:

1. *extracting-data-using-python.ipynb* - contains code for extracting the data from the Kaggle website
2. *processing-data-using-python-1.0.ipynb* - contains the code for cleaning & processing train data to build a predictive model
3. *processing-data-using-python-1.1.ipynb* - contains the code for cleaning & processing train data to build a predictive model
4. *predictive-model-using-python.ipynb* - contains code to build baseline model and logistic regression predictive model

Validation:

For validating the final predictive model, here we are creating a Baseline predictive model and comparing the final Logistic Regression model with the Baseline model.

For the Baseline model, we have the accuracy of around 0.61 ~ 61%

```
In [36]: print('score for baseline model : {0:.2f}'.format(model_dummy.score(X_test, Y_test)))

score for baseline model : 0.61

In [37]: # performance metrics
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score, recall_score

In [40]: # accuracy score
print('accuracy for baseline model : {0:.2f}'.format(accuracy_score(Y_test, model_dummy.predict(X_test))))

accuracy for baseline model : 0.61
```

For the Logistic Regression model, we have accuracy more than the Baseline model ie. 0.83 ~ 83%

```
The score for logistic regression model 0.8324022346368715

In [53]: # performance metrics
# accuracy
print('accuracy for logistic regression - version 1 : {0:.2f}'.format(accuracy_score(Y_test, log_rg_model_v1.predict(X_test))))
# confusion matrix
print('confusion matrix for logistic regression - version 1: \n {0}'.format(confusion_matrix(Y_test, log_rg_model_v1.predict(X_test))))
# precision
print('precision for logistic regression - version 1 : {0:.2f}'.format(precision_score(Y_test, log_rg_model_v1.predict(X_test))))
# precision
print('recall for logistic regression - version 1 : {0:.2f}'.format(recall_score(Y_test, log_rg_model_v1.predict(X_test))))

accuracy for logistic regression - version 1 : 0.83
confusion matrix for logistic regression - version 1:
[[95 15]
 [15 54]]
precision for logistic regression - version 1 : 0.78
recall for logistic regression - version 1 : 0.78
```

Also, we are validating our model wrt Kaggle where we have the actual survival rate

Overview

Data

Notebooks

Discussion

Leaderboard

Rules

Team

My Submissions

Submit Pr

117...	dias711		0.77033
117...	U.ryosuke		0.77033
117...	rajeshwar-sharma		0.77033

Your Best Entry ↑

You advanced 6,168 places on the leaderboard!