Project Title

Machine Learning Recipes for Predictive Model Development with Python

Developed by: MSc. Shiva Agrawal

Place: Germany

Date: September 2018

Content

1. Introduction

1.1. Project description

The aim of this project work is to collect and document all the steps required to develop the predictive model using machine learning. This process involves steps like importing the data into environment like Python, C++, etc., analyzing the dataset by generating different statistical results and plots for visualization, data preprocessing, feature selection, cross validation types, accuracy metrics selection, etc.

After having good knowledge about the dataset and its different parameters and after cleaning and processing the same, different ML algorithms are tried and tested to find the best fit. Moreover, after deciding the best fit model, it is important to save the developed model to use it for future.

As all these steps are mostly same for working on different ML predictive model development project, it is always good if everything is available properly in form of recipes which can be easily used by the developer for the project quickly.

In this project, **python scikit-learn package** is used for the machine learning and **pandas** to import the dataset into python environment in form of **dataframe.**

The project is developed by referring the book Mastering Machine Learning using Python by Jason Brownlee [1], scikit-learn official website and other online references.

1.2. Outline

- Chapter 1: Project description and outline of the project report
- Chapter 2: Python packages information and dataset(s)
- Chapter 3: Machine Learning recipes
- Chapter 4: Conclusion
- Chapter 5: References to other websites, technical papers referred, etc.

2. Python packages and datasets

2.1. Python Packages

2.1.1. Scikit-learn

[2] This is open source package available for Machine Learning in Python. It is built on python other packages like numpy, scipy and matplotlib. It contains most of the required functions and tools to preprocess, analyze and develop the ML models.

2.1.2. Pandas

[3] It is an open source, BSD License library providing high performance, easy to use data structure and data analysis tool for the python programming language.

2.1.3. Numpy

[4] It is the fundamental package for scientific computing in python. It has powerful N dimension array object, sophisticated broadcasting functions and useful linear algebra, Fourier transform, and random number capabilities.

2.1.4. Scipy

[5] SciPy is a free and open-source Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.

2.1.5. Matplotlib

[6] Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. It can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc.

2.2. Datasets

2.2.1. Pima Indian Diabetes Data

[7] It contains the dataset with 8 features and 1 outcome. This is binary classification dataset.

Features and outcome are as below.

1. Number of times pregnant
2. Plasma glucose concentration of 2 hours in an oral glucose tolerance test
3. Diastolic blood pressure (mm Hg)
4. Triceps skin fold thickness (mm)
5. 2-Hour serum insulin (mu U/ml)
6. Body mass index (weight in kg/ (height in m) ^2)
7. Diabetes pedigree function

8. Age (years)
9. Class variable (0 or 1)

This dataset contains total 768 samples.

2.2.2. Boston House price data

[8] It contains the dataset with 13 features and 1 output. This is for regression type problems.

Features and outcome are as below:

1. CRIM	per capita crime rate by town
2. ZN	proportion of residential land zoned for lots over 25,000 sq. Ft.
3. INDUS	proportion of non-retail business acres per town
4. CHAS	Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
5. NOX	nitric oxides concentration (parts per 10 million)
6. RM	average number of rooms per dwelling
7. AGE	proportion of owner-occupied units built prior to 1940
8. DIS	weighted distances to five Boston employment centers
9. RAD	index of accessibility to radial highways
10. TAX	full-value property-tax rate per $10,000
11. PTRATIO	pupil-teacher ratio by town
12. B	1000(Bk - 0.63) ^2 where Bk is the proportion of blacks by town
13. LSTAT	% lower status of the population
14. HOUSE_PRICE	Median value of owner-occupied homes in $1000's

This dataset contains 506 samples.

3. Machine Learning Recipes

This section provides recipes overview which includes dataset import, preprocessing, ML model development algorithms, saving and loading developed models, etc. The source code for each subsection is available in src folder of the repository under each sub folder.

NOTE: Please first copy the required datasets into the corresponding folder from the src folder and then use the recipes.

3.1. Load CSV data into Python

Different ways to load CSV file into Python environment are

1. using python standard library
2. using Numpy
3. Using Pandas - Dataframe

Source code is available at: **src/01 Import Dataset**

3.2. Data analysis

After importing the dataset, the very first thing is to understand and analyze the data and its different parameters/features and their relations. This can be done by finding statistical values of the data and by visualizing the different features using various plots.

Statistics covered for the dataset are:

- peek / look at raw data.
- Dimensions of your data
- Data types of attributes of the data.
- Summarize the distribution of instances across classes in dataset
- Summarize data using descriptive statistics like mean, median, variance, etc.
- Understand the relationships in data using correlations
- Skew of the distributions of each attribute

Visualization plots are:

Univariate plots	Histograms Density Plots Box and Whisker plots
Multivariate plots	Correlation matrix plot Scatter plot Matrix

Source code is available at: **src/02 Data Analysis**

3.3. Data pre-processing

Many machine learning algorithms make assumptions about data and It is often a very good idea to prepare the data in such way to best expose the structure of the problem to the machine learning algorithms that one intends to use.

The four methods of pre-processing the data are:

- Rescale Data
- Standardize Data
- Normalize Data
- Binarize Data

Each method above follow the same steps as:

1. Load the Dataset from URL or other function
2. Split the Dataset into input and output variables for ML
3. Apply one or more pre-processing transform (as stated above) to the input variables
4. Summarize the data to show the change

Scikit-learn library of Python provides two standard idioms for transforming data

1. Fit and Multiple Transform (using fit (), and transform () functions)
2. Combined fir and transform (using fit_transform () function)

Source code is available at: **src/03 Data Pre-Processing**

3.4. Feature Selection

Feature selection is a process where we automatically select those features in the data that contribute most to the prediction variable or output in which we are interested. Having irrelevant features in the data can decrease the accuracy of many models, especially linear algorithms like linear and logistic regression.

Benefits of feature selection

1. Reduce Overfitting
2. Improve model accuracy
3. Reduce training time

The automatic feature selection techniques are:

1. Univariate selection
2. Recursive Feature Elimination
3. Principle Component Analysis (PCA)
4. Feature Importance

Source code is available at: **src/04 Feature Selection**

3.5. Performance Evaluation with cross validation

When we have a dataset (let’s say of 768 samples of Pima Indian diabetic dataset) and we used it and trained one ML Model.

Now we get some new samples (not from used 768) and want to use the developed model to get output y. But it is found that the result is not as expected. But if we use one of the samples from used 768, then developed model provides quite correct output.

This is the result of overfitting. Hence, we cannot use the trained data for performance evaluation and if we don't do performance evaluation then model may not work for new samples as per requirements.

Following are the methods can be used to do Performance evaluation with cross validation

1. Train and Test datasets
2. k-fold Cross Validation
3. Leave One out Cross Validation
4. Repeat Random Test-Train splits

What technique to use when?

- K-fold cross validation (cv) is the gold standard for evaluating the performance of a machine learning algorithm on unseen data with k = 3,5, 10, etc.
- Using a train/test split is good for speed when using a slow algorithm and produces performance estimates with lower bias when using large datasets.
- Techniques like leave-one-out cross validation and repeated random splits can be useful intermediates when trying to balance variance in the estimated performance, model training speed and dataset size.
- The best advice is to experiment and find a technique for your problem that is fast and produces reasonable estimates of performance that you can use to make decisions. If in doubt, use 10-fold cross validation.

Source code is available at: **src/05 Cross Validation**

3.6. Performance Metrics

After development of the model and cross validation, it is important to define a certain way to tell whether the developed model is able to fulfill our criteria of accuracy. Such definitions come under metrics. The metrics that you choose to evaluate your machine learning algorithms are very important.
Choice of metrics influences how the performance of machine learning algorithms is measured and compared. Depending on type of model, metrics can be different and also the interpretation of their results.

Classification Metrics

- Classification Accuracy
- Logarithmic Loss
- Area under ROC Curve
- Confusion Matrix
- Classification Report

Regression Metrics

- Mean Absolute Error
- Mean Square Error
- R2 (R square)

Source code is available at: **src/06 Performance Metrics**

3.7. ML Classification Algorithms

Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y). Classification belongs to the category of supervised learning where the targets also provided with the input data.

When the output of the dataset is specific (descrete valued) then we develop classification model. If outputs are only 2 then it is called as Binary Classification / Logistic Regression.

Following linear and non-linear classification algorithms are covered.

Linear:

- Logistic Regression
- Linear Discriminant Analsis

Non-linear:

- K-Nearest Neighbour (KNN)
- Naive Byes
- Classification and Regression Trees
- Support Vector Machine (SVM)

- Each of the above method is implemented on Pima Indian Diabetic Dataset (768 x 9).
- For performing cross validation, k fold CV is used with k = 10
- For evaluating the performance, classification accuracy metrics is used

General steps to develop a model,

1. model = LogisticRegression() # example - create its instance
2. model.fit(input, output) # fit the training data
3. Do cross validation
4. model.predict(test_samples) # to find new outputs

Source code is available at: **src/07 Classification Algorithms**

3.8. ML Regression Algorithms

Regression, one of the most common types of machine learning models, estimates the relationships between variables. Regression is essential for any machine learning problem that involves continuous numbers, which includes a vast array of real-life applications.

Following linear and non-linear regression algorithms are covered

Linear:

- Linear Regression
- Ridge Regression
- LASSO Linear Regression
- Elastic net Regression

Non-linear:

- K-Nearest Neighbour (KNN)
- Classification and Regression Trees
- Support Vector Machine (SVM)

- Each of the above method is implemented on Boston House Price Dataset.
- For performing cross validation, k fold CV is used with k = 10
- For evaluating the performance, Regression (R2) and Mean Squared Error metrics are used

General steps to develop a model,

5. model = LinearRegression() # example - create its instance
6. model.fit(input, output) # fit the training data
7. Do cross validation
8. model.predict(test_samples) # to find new outputs

Source code is available at: **src/08 Regression Algorithms**

3.9. Compare algorithms

After development of model using different ML Algorithms, it is important to analyze the results consistently and decide which one or two models best fit the requirements. We can do this by using visualization methods to show the average accuracy, variance and other properties of the distribution of model accuracies.

The key to a fair comparison of machine learning algorithms is ensuring that each algorithm is evaluated in the same way on the same data. This can be achieved by forcing each algorithm to be evaluated on a consistent test method.

Source code is available at: **src/09 Compare Algorithms**

3.10. Save and Load ML model(s)

After development of the final ML Model, it is required to save the model for use either in other project or to use in future. There are many different ways to save the models depending on which type of libraries are used.

Moreover, there are two categories of model to save

1. Develop and save model in Python and load in Python (described in this section)
2. Develop model in Python but save it in other languages (C/C++) for further production code generation (this is out of scope of this project)

Two packages are used to save and load the data into Python

1. Pickle
2. Joblib

Source code is available at: **src/10 Save and Load Model**

4. Test and results

Since the project is collection of recipes where each source code is self-explanatory and the output can be obtained just by running the corresponding recipe, there are no specific tests or results required

5. Conclusion

Recipes for all the steps to develop ML predictive model from data import to save the final model are implemented and tested in Python with Ubuntu 16.04.

Bibliography

[1] J. Brownlee, Mastering Machine Learning using Python.

[2] "Home page," [Online]. Available: http://scikit-learn.org/stable/.

[3] "pandas," [Online]. Available: https://pandas.pydata.org/.

[4] "Numpy," [Online]. Available: http://www.numpy.org/.

[5] "scipy," [Online]. Available: https://www.scipy.org/.

[6] "Maplotlib," [Online]. Available: https://matplotlib.org/.

[7] "pima-indians-diabetes-database," kaggle, [Online]. Available: https://www.kaggle.com/uciml/pima-indians-diabetes-database.

[8] "UCI repository archieve," UCI, [Online]. Available: https://archive.ics.uci.edu/ml/machine-learning-databases/housing/.