MODULE 08 - Piscine Python for Data Science

Intro to Machine Learning

Summary: Today we will help you with basic tasks involved in machine learning in Python.

Contents

Chapter I

Foreword

- There are a lot of different terms that are connected to the data field: artificial intelligence, machine learning, neural nets, and deep learning. But do you know the difference between them?
- Each item of the list is a subset of the previous item. The broadest term is artificial intelligence includes any techniques that mimic human cognitive behavior. It can use machine learning algorithms in order to do this, or any other techniques like just writing a program with many "if-then-else" rules.
- Machine learning includes statistical algorithms that automate the process of creating the rules. The machine can find correlations and use them for different tasks by "looking" at the data.
- Neural nets are a subset of machine learning algorithms. They were created through the inspiration of how the human brain works (but they are still a far cry away from it). And deep learning algorithms are a subset of neural nets. They usually have many layers. That is why they are called "deep".
- Remember that none of these algorithms are limitless. They can help you only if you have the data. Simple algorithms can be satisfied with small amounts of data, and deep learning algorithms require large amounts of data. At the same time, if your data is garbage, the knowledge that you get from the algorithms will be garbage too. No surprise, huh?

Chapter II

Instructions

- Use this page as your only reference. Do not listen to any rumors or speculations about how to prepare your solution.
- Here and further on we use Python 3 as the only correct version of Python.
- The solutions for python exercises (d01, d02, d03) must have the following block in the end: if ___name___ == '___main___'.
- Pay attention to the permissions of your files and directories.
- To be assessed your solution must be in your GIT repository.
- Your solutions will be evaluated by your piscine peers.
- You should not leave any additional files in your directory other than those explicitly specified in the subject. It is recommended that you modify your .gitignore to avoid any accidents.
- When you need to get precise output in your programs, it is forbidden to display a precalculated output instead of performing the exercise correctly.
- Have a question? Ask your neighbor on the right. If that fails, try your neighbor on the left.
- Your reference material: peers / Internet / Google.
- You can ask questions in Slack.
- Read the examples carefully. They may require things that are not otherwise specified in the subject.
- And may the Force be with you!

Chapter III

Specific instructions of the day

- Use Jupyter Notebook to work with your code
- For each major subtask in the list of any exercise (black bullets), your ipynb file should have an h2 heading to help your peer easily navigate within your code
- No imports are allowed, except those explicitly mentioned in the section "Authorized functions" of the title block of each exercise
- You can use any built-in function, as long as it is not prohibited in the exercise
- Save and load all the required data to the subfolder data/
- scikit-learn (0.23.1) is the library that you need for all the machine learning tasks.

Chapter IV

Exercise 00 : Binary classifier

	Exercise 00
Binary classifier	
Turn-in directory : *ex*00/	
Files to turn in : `00_binary_classifier_logreg.ipynb`	
Allowed functions : `no restrictions`	

You will work with machine learning today and the following day. Today we will cover the basics necessary to not to get overwhelmed by too much information. The following day, will cover some more sophisticated techniques. So stay tuned!

First of all, machine learning can be divided into supervised and unsupervised. In supervised learning, you want to predict something. In order to do that, you give the machine examples: a bunch of features and a target variable. Imagine that you want to predict whether a user would like or dislike a given movie. The features (X) can be: genre, year of creation, budget, cast, director, and so on. The target (y) will be like/dislike. That kind of task is a classification task. In classification problems, y is always categorical. If your target variable is continuous (for instance, a movie rating), it is called a regression problem.

Unsupervised learning does not require labels (target variable). It does not forecast anything. Usually, it helps you to understand your data better. For example, clustering algorithms help you identify homogeneous groups of observations. You may find that you can divide your users into 6 groups. And for each of these groups create a special offer or a recommendation. Do not confuse this with classification. You are not trying to predict anything. You are just looking at your data.

Besides clustering algorithms, unsupervised learning includes dimensionality reduction algorithms. They help you to reduce the number of features or observations. It may be helpful if you have a lot of them, but you do not have sufficient resources. Also, they may help you to find some latent features and improve the quality of your algorithms in supervised learning.

But enough theory! Let us try to train our first classifier. It will be a binary classifier which means that the target variable has only two unique values. You will work with the dataset from the previous days. It is absolutely normal. Actually, that is how data science works. You start from the descriptive analysis and proceed by analyzing some basic statistics. Then you go further and do explorative analysis by drawing different plots and, as a result, get a better understanding of your data. Only then do you move on to predictive analysis to forecast something.

In this exercise imagine the following situation (which is, by the way, quite common). From some moment in the past, you realized that the more data you collect, the better. And you started saving more fields in the logs. Before, you had been collecting only the time of the commit. But from that point in time, you started collecting the date too.

Now you need to train a classifier that can predict whether any given commit was made during working days or during weekends. Then you will be able to use the classifier to label the commits in the past when you had not been collecting the data.

Every supervised machine learning algorithm requires at least two arguments: X and y. X is the list of features and y is the target column. But what are the features?

As you no doubt recall, we only have logs like this 2020-04-17 05:19:02.744528. How can we use this to predict the type of weekday? That is the creative part of the work. It is called feature engineering. You need to extract these features from the logs. What could it be? Remember that the observation in this task is a day? So, we need to extract something that can somehow characterize days. What could it be? It may be, for example, the number of commits during the day. It may be the number of commits before midday and afterward. Or it may be also the percentage of commits made before midday. You are only limited by your imagination!

In this exercise, we will try a simple approach with only two features: the number of commits before midday and the number of commits after midday.

- What you need to do is described in full details in the notebook.

Chapter V

Exercise 01 : Decision boundaries

	Exercise 01
Decision boundaries	
Turn-in directory : *ex*01/	
Files to turn in : `01_binary_decision_boundaries.ipynb`	
Allowed functions : `no restrictions`	

Ok, you trained your first classifier! Now it seems probably like magic to you. Let us look under the hood, try to increase the quality of your classifier, and try other algorithms.

In this exercise you will see how logistic regression works. Also, you will try two more machine learning algorithms: SVM and decision tree. You will visualize them too.

- What you need to do is described in full detail in the notebook.

Chapter VI

Exercise 02 : Multiclass

	Exercise 02
Multiclass	
Turn-in directory : *ex*02/	
Files to turn in : `02_multiclassi_one-hot.ipynb`	
Allowed functions : `no restrictions`	

Ok, now you understand more or less how different algorithms make classifications. It is time to get closer to a real life scenario. In real life, your target column may contain more than two values. In such cases, you will need to train not a binary classifier, but a multiclass classifier (do not confuse it with multilabel – we're talking when your observations can belong to several classes at the same time).

Also, you may have not only continuous features but categorical as well. You need to deal with them somehow. Algorithms understand numbers, they do not know what to do with text.

In this exercise, you will work with both problems. Also, you will find out which features seem the most important for different algorithms. You will also try one more algorithm random forest.

What you need to do:

- What you need to do is described in full detail in the notebook.

Ain't it cool that we can predict the weekday of any commit with high accuracy knowing who made it, at what time, for which lab and how many tries they had already made?

Chapter VII

Exercise 03 : Overfitting

	Exercise 03
Overfitting	
Turn-in directory : *ex*03/	
Files to turn in : `03_split_crossval.ipynb`	
Allowed functions : `no restrictions`	

We are sure that you managed to achieve pretty high values of accuracy. But those numbers are kind of unfair. Why? You were making predictions on the same data that you had used for training. Your models could just memorize all the observations. In theory, you could achieve 100% accuracy, but would your model still be any good if it tried to make predictions for data that it had not seen? We highly doubt it. That is called overfitting.

One of the techniques to prevent it is to make a train/test split. You get a portion of data that you use for training, and another portion you use to check the final quality of your model. The second is cross-validation. In this technique, we do not make a constant split, but we try different splits and see what quality of predictions we get.

- What you need to do is described in full detail in the notebook.

Chapter VIII

Exercise 04 : Regression

	Exercise 04
Regression	
Turn-in directory : *ex*04/	
Files to turn in : `04_regression.ipynb`	
Allowed functions : `no restrictions`	

You now know a thing or two about classification tasks. Let us work on a regression problem in this exercise. You will need to predict the average delta between the deadlines and the first commit for every user with data about their views of the newsfeed and the number of commits they made during the program.

- What you need to do is described in full detail in the notebook.

Chapter IX

Exercise 05 : Clustering

	Exercise 05
Clustering	
Turn-in directory : *ex*05/	
Files to turn in : `05_clustering.ipynb`	
Allowed functions : `no restrictions`	

It is time to try using unsupervised machine learning. This time we will work with clustering algorithms. We will try to understand whether we can divide our users into some homogenous groups for future analysis. Maybe we can add some more triggers or engaging mechanics for them. But the triggers and mechanics may differ depending on the users' existing behavior.

- What you need to do is described in full detail in the notebook.