LinearUCB

Exercise Lecture

Contents

Motivations

When the arms space is huge we can exploit the information gained on observed arms to estimate the reward function of non-observed arms

Linear Environment

Each arm j is associated with a feature vector $x_j = \left(x_{j1}, x_{j2}, x_{j3}, \ldots, x_{jD} \right) with\ x_{ji} \in [0,1]$

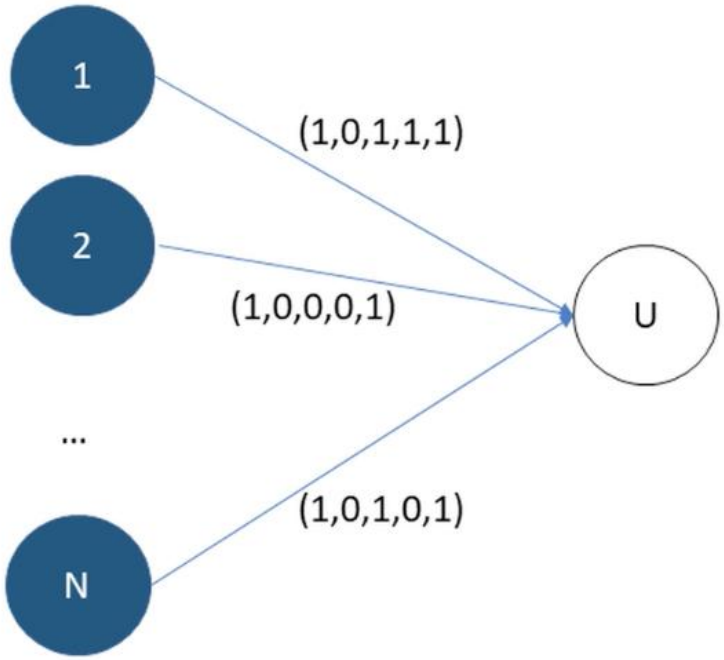
Linear Environment

Each arm j is associated with a feature vector $x_j = \left(x_{j1}, x_{j2}, x_{j3}, \dots, x_{jD} \right) with\ x_{ji} \in [0,1]$

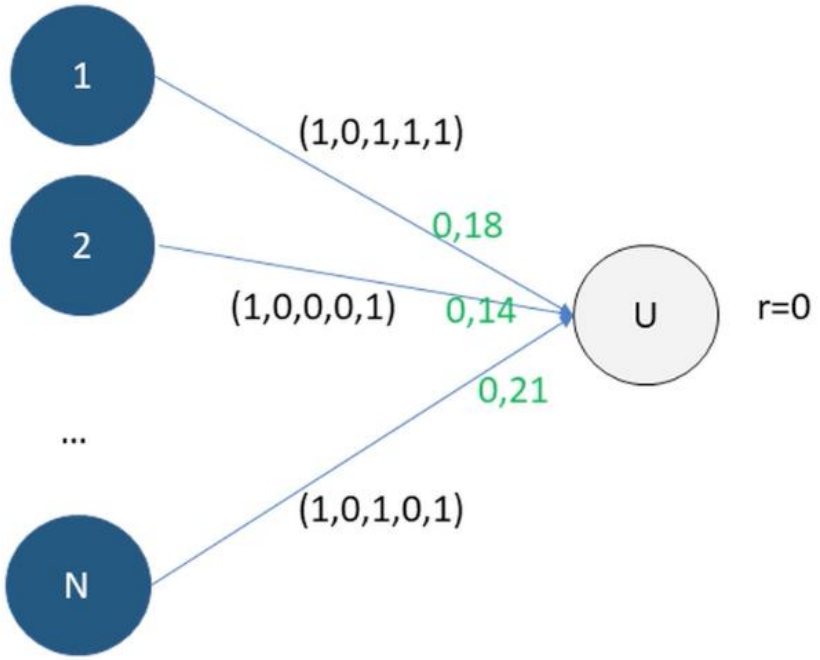
The reward is a linear combination of the arm feature vector and a parameters vector θ:

$$r_t = x_t^T \theta$$

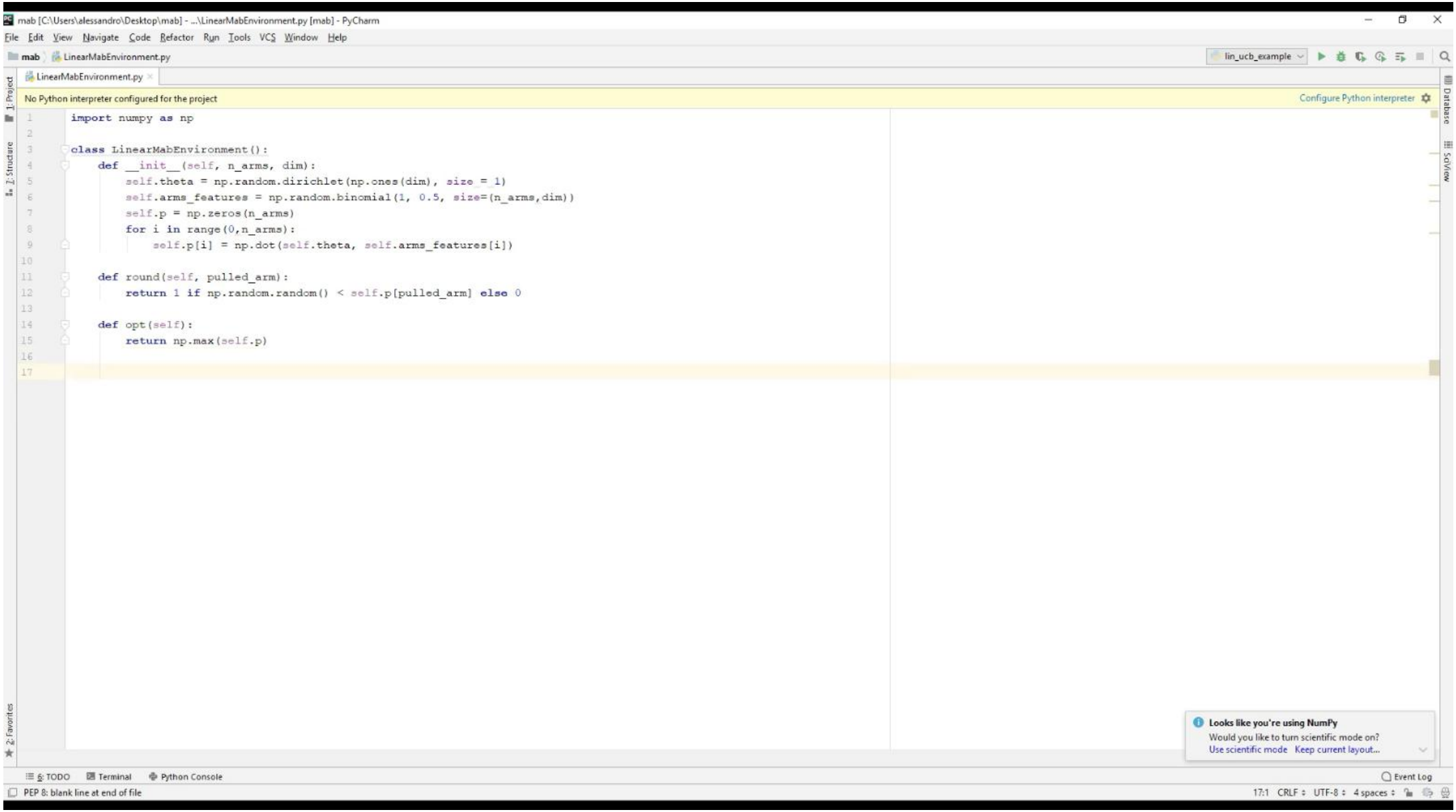
Example: Social Influence



Example: Social Influence



Let’s implement it!


mab [C:\Users\elessandro\Desktop\mab] - ...\LinearMabEnvironment.py [mab] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
mab LinearMabEnvironment.py
lin_ucb_example
LinearMabEnvironment.py
No Python interpreter configured for the project
Configure Python interpreter
import numpy as np
class LinearMabEnvironment():
def __init__(self, n_arms, dim):
self.theta = np.random.dirichlet(np.ones(dim), size = 1)
self.arms_features = np.random.binomial(1, 0.5, size=(n_arms,dim))
self.p = np.zeros(n_arms)
for i in range(0,n_arms):
self.p[i] = np.dot(self.theta, self.arms_features[i])
def round(self, pulled_arm):
return 1 if np.random.random() < self.p[pulled_arm] else 0
def opt(self):
return np.max(self.p)
Looks like you're using NumPy
Would you like to turn scientific mode on?
Use scientific mode Keep current layout...
6: TODO
Terminal
Python Console
Event Log
PEP 8: blank line at end of file
17:1 CRLF UTF-8 4 spaces
1: Project
7: Structure
2: Favorites
Database
SciView

LinearUCB

Input: arms set A, parameter c>0

Initialization: $B_0 = 0 \in R^d, M_0 = I \in R^{dxd}$

LinearUCB

Input: arms set A, parameter c>0

Initialization: $B_0 = 0 \in R^d, M_0 = I \in R^{dxd}$

For $t = 1,2,\dots,n$ do

1) $\theta_{t-1} = M_{t-1}^{-1} B_{t-1}$ a

UCBs = $x^T \theta_{t-1} + c\sqrt{x^T M_{t-1}^{-1} x}$

LinearUCB

Input: arms set A, parameter c>0

Initialization: $B_0 = 0 \in R^d, M_0 = I \in R^{dxd}$

For $t = 1,2,\dots,n$ **do**

1) $\theta_{t-1} = M_{t-1}^{-1} B_{t-1}$ a

UCBs = $x^T \theta_{t-1} + c\sqrt{x^T M_{t-1}^{-1} x}$

2) Choose the arm with maximum ucb value

LinearUCB

Input: arms set A, parameter c>0

Initialization: $B_0 = 0 \in R^d, M_0 = I \in R^{dxd}$

For $t = 1,2, \dots, n$ do

1) $\theta_{t-1} = M_{t-1}^{-1} B_{t-1}$ a

UCBs = $x^T \theta_{t-1} + c\sqrt{x^T M_{t-1}^{-1} x}$

2) Choose the arm with maximum ucb value

3) Update matrices:

a) $M_t = M_{t-1} \quad B_t = B_{t-1}$

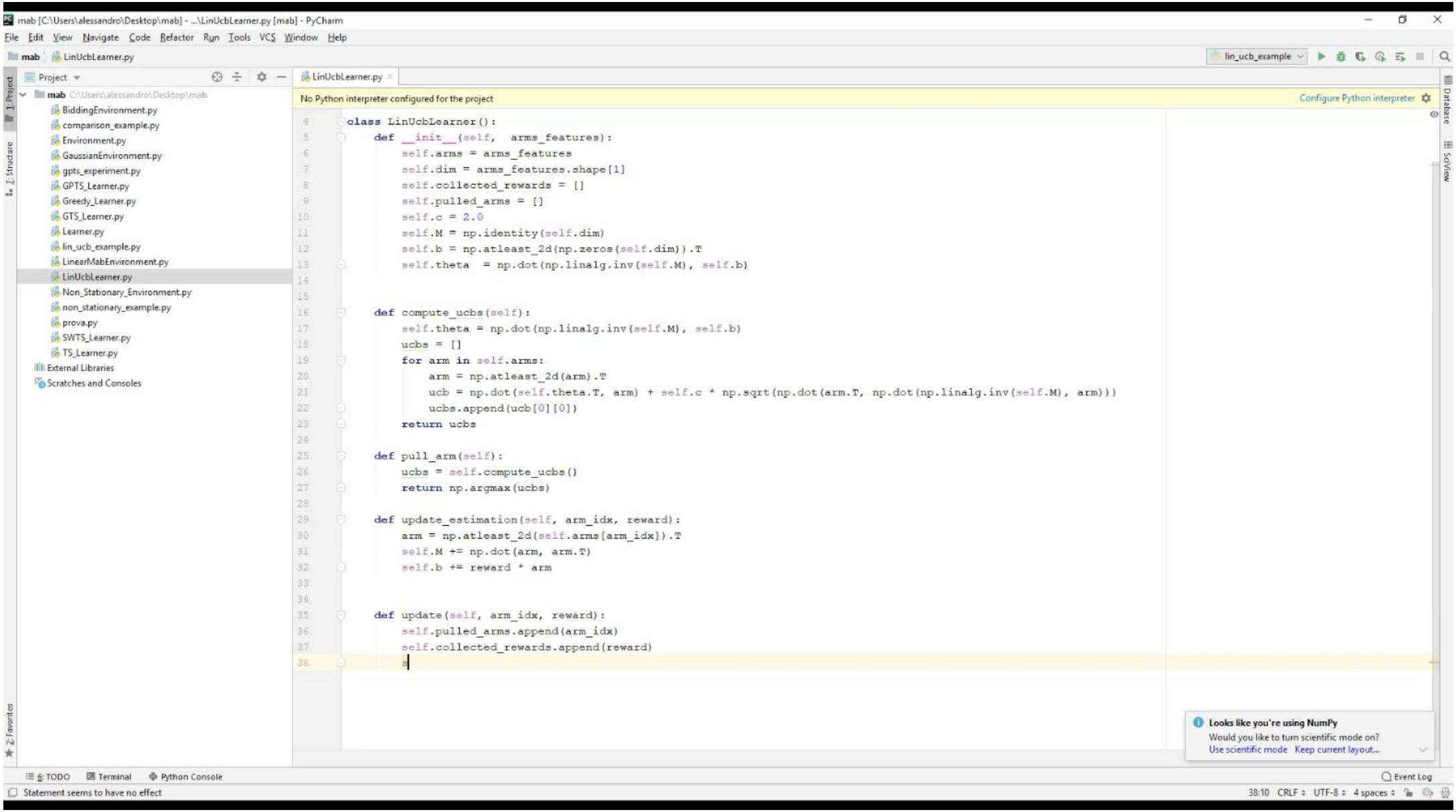
b) $M_t = M_t + x_t x_t^T$

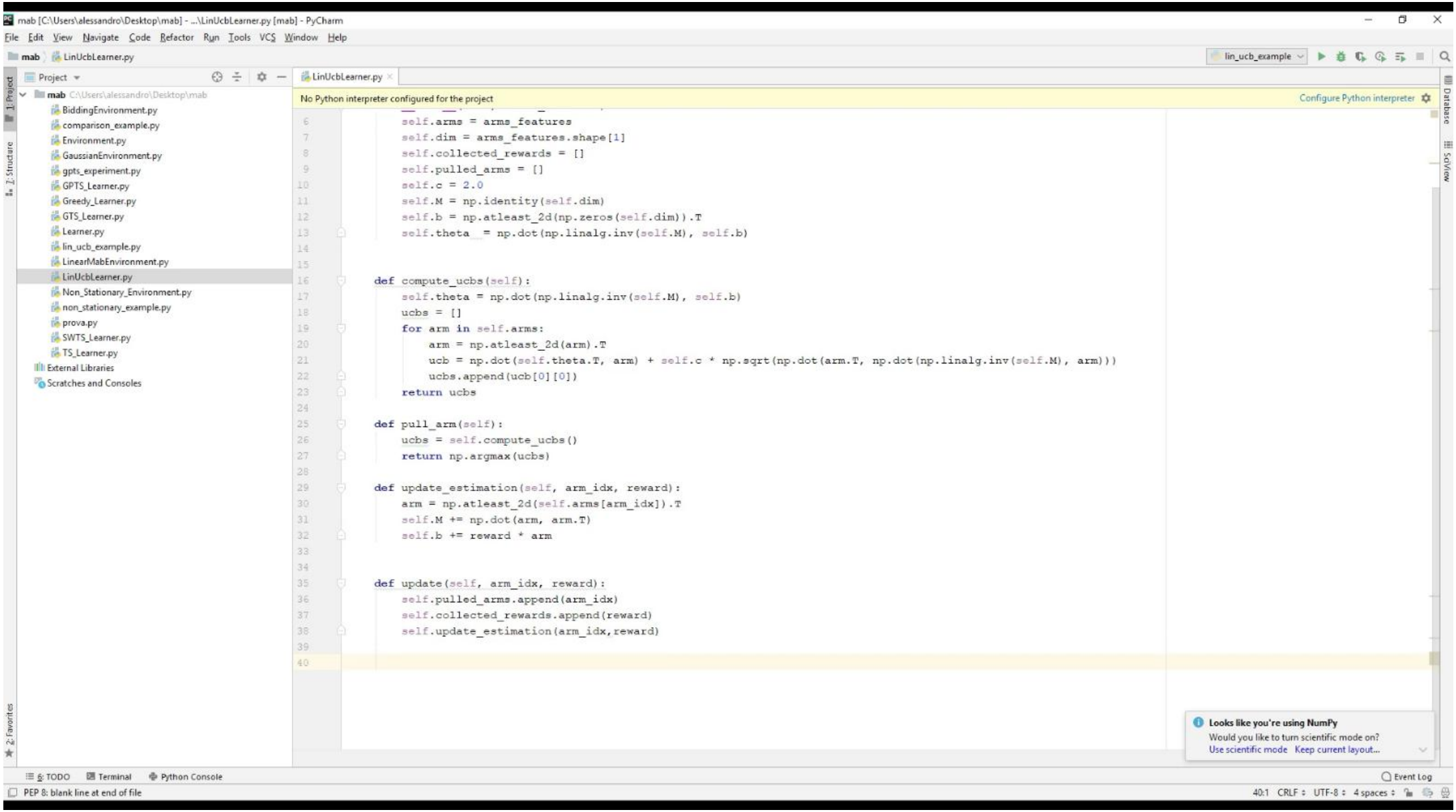
$and\ B_t = B_t + x_t r_t$

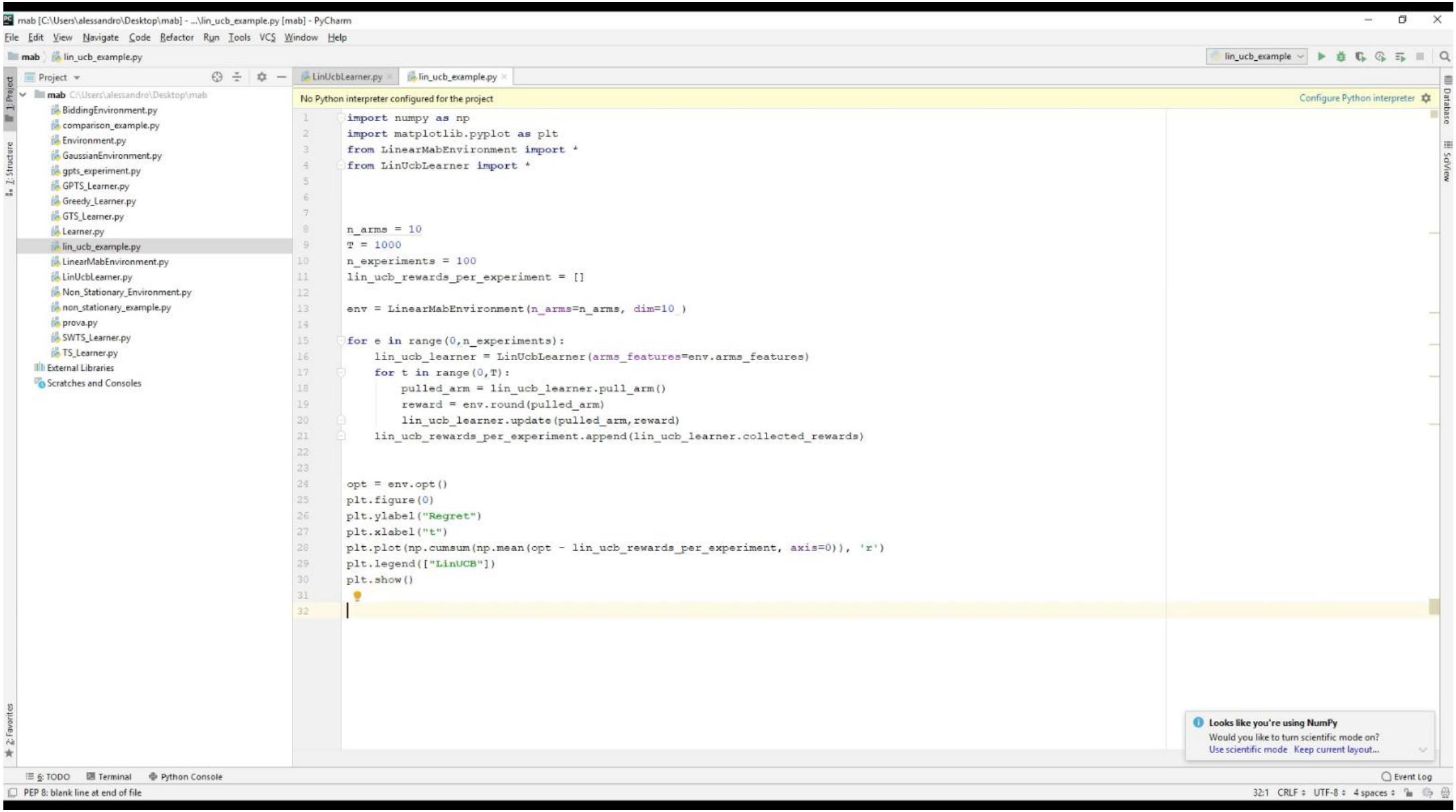
IMLinUCB

Wen, Zheng, et al. "**Online influence maximization under independent cascade model with semi-bandit feedback.**" *Advances in neural information processing systems*. 2017.

Let’s implement it!


mab [C:\Users\alessandro\Desktop\mab] - ...\LinUcbLearner.py [mab] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
mab LinUcbLearner.py
lin_ucb_example
Project
mab C:\Users\alessandro\Desktop\mab
BiddingEnvironment.py
comparison_example.py
Environment.py
GaussianEnvironment.py
gpts_experiment.py
GPTS_Learner.py
Greedy_Learner.py
GTS_Learner.py
Learner.py
lin_ucb_example.py
LinearMabEnvironment.py
LinUcbLearner.py
Non_Stationary_Environment.py
non_stationary_example.py
prova.py
SWTS_Learner.py
TS_Learner.py
External Libraries
Scratches and Consoles
LinUcbLearner.py
No Python interpreter configured for the project
Configure Python interpreter
class LinUcbLearner():
def __init__(self, arms_features):
self.arms = arms_features
self.dim = arms_features.shape[1]
self.collected_rewards = []
self.pulled_arms = []
self.c = 2.0
self.M = np.identity(self.dim)
self.b = np.atleast_2d(np.zeros(self.dim)).T
self.theta = np.dot(np.linalg.inv(self.M), self.b)
def compute_ucbs(self):
self.theta = np.dot(np.linalg.inv(self.M), self.b)
ucbs = []
for arm in self.arms:
arm = np.atleast_2d(arm).T
ucb = np.dot(self.theta.T, arm) + self.c * np.sqrt(np.dot(arm.T, np.dot(np.linalg.inv(self.M), arm)))
ucbs.append(ucb[0][0])
return ucbs
def pull_arm(self):
ucbs = self.compute_ucbs()
return np.argmax(ucbs)
def update_estimation(self, arm_idx, reward):
arm = np.atleast_2d(self.arms[arm_idx]).T
self.M += np.dot(arm, arm.T)
self.b += reward * arm
def update(self, arm_idx, reward):
self.pulled_arms.append(arm_idx)
self.collected_rewards.append(reward)
Looks like you're using NumPy
Would you like to turn scientific mode on?
Use scientific mode Keep current layout...
6: TODO
Terminal
Python Console
Event Log
Statement seems to have no effect
38:10 CRLF UTF-8 4 spaces


mab [C:\Users\alessandro\Desktop\mab] - ...\LinUcbLearner.py [mab] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
mab LinUcbLearner.py
lin_ucb_example
Project
mab C:\Users\alessandro\Desktop\mab
BiddingEnvironment.py
comparison_example.py
Environment.py
GaussianEnvironment.py
gpts_experiment.py
GPTS_Learner.py
Greedy_Learner.py
GTS_Learner.py
Learner.py
lin_ucb_example.py
LinearMabEnvironment.py
LinUcbLearner.py
Non_Stationary_Environment.py
non_stationary_example.py
prova.py
SWTS_Learner.py
TS_Learner.py
External Libraries
Scratches and Consoles
LinUcbLearner.py
No Python interpreter configured for the project
Configure Python interpreter
self.arms = arms_features
self.dim = arms_features.shape[1]
self.collected_rewards = []
self.pulled_arms = []
self.c = 2.0
self.M = np.identity(self.dim)
self.b = np.atleast_2d(np.zeros(self.dim)).T
self.theta = np.dot(np.linalg.inv(self.M), self.b)
def compute_ucbs(self):
self.theta = np.dot(np.linalg.inv(self.M), self.b)
ucbs = []
for arm in self.arms:
arm = np.atleast_2d(arm).T
ucb = np.dot(self.theta.T, arm) + self.c * np.sqrt(np.dot(arm.T, np.dot(np.linalg.inv(self.M), arm)))
ucbs.append(ucb[0][0])
return ucbs
def pull_arm(self):
ucbs = self.compute_ucbs()
return np.argmax(ucbs)
def update_estimation(self, arm_idx, reward):
arm = np.atleast_2d(self.arms[arm_idx]).T
self.M += np.dot(arm, arm.T)
self.b += reward * arm
def update(self, arm_idx, reward):
self.pulled_arms.append(arm_idx)
self.collected_rewards.append(reward)
self.update_estimation(arm_idx,reward)
Looks like you're using NumPy
Would you like to turn scientific mode on?
Use scientific mode Keep current layout...
6: TODO
Terminal
Python Console
Event Log
PEP 8: blank line at end of file
40:1 CRLF UTF-8 4 spaces
1: Project
7: Structure
2: Favorites
Database
SciView


mab [C:\Users\alessandro\Desktop\mab] - ...\lin_ucb_example.py [mab] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
mab lin_ucb_example.py
lin_ucb_example
Project
mab C:\Users\alessandro\Desktop\mab
BiddingEnvironment.py
comparison_example.py
Environment.py
GaussianEnvironment.py
gpts_experiment.py
GPTS_Learner.py
Greedy_Learner.py
GTS_Learner.py
Learner.py
lin_ucb_example.py
LinearMabEnvironment.py
LinUcbLearner.py
Non_Stationary_Environment.py
non_stationary_example.py
prova.py
SWTS_Learner.py
TS_Learner.py
External Libraries
Scratches and Consoles
LinUcbLearner.py
lin_ucb_example.py
No Python interpreter configured for the project
Configure Python interpreter
import numpy as np
import matplotlib.pyplot as plt
from LinearMabEnvironment import *
from LinUcbLearner import *
n_arms = 10
T = 1000
n_experiments = 100
lin_ucb_rewards_per_experiment = []
env = LinearMabEnvironment(n_arms=n_arms, dim=10)
for e in range(0,n_experiments):
lin_ucb_learner = LinUcbLearner(arms_features=env.arms_features)
for t in range(0,T):
pulled_arm = lin_ucb_learner.pull_arm()
reward = env.round(pulled_arm)
lin_ucb_learner.update(pulled_arm,reward)
lin_ucb_rewards_per_experiment.append(lin_ucb_learner.collected_rewards)
opt = env.opt()
plt.figure(0)
plt.ylabel("Regret")
plt.xlabel("t")
plt.plot(np.cumsum(np.mean(opt - lin_ucb_rewards_per_experiment, axis=0)), 'r')
plt.legend(["LinUCB"])
plt.show()
Looks like you're using NumPy
Would you like to turn scientific mode on?
Use scientific mode Keep current layout...
6: TODO
Terminal
Python Console
Event Log
PEP 8: blank line at end of file
32:1 CRLF UTF-8 4 spaces
1: Project
7: Structure
2: Favorites
Database
SciView