Profession recommendations

Akmurat Ulzhan
Sovetbek Zhansaya
01/12/2019

Abstract

This documentation is all about applying machine learning solutions for real practical use cases. So the focus is on outlining how we collect data from several sources and machine learning algorithms.

Machine learning is an exciting powerful technology. The continuous use and growth of machine learning technology opens new opportunities. It should also give improvements for everyone. Our project solves the problem of people: becoming a well-rounded individual. People try different activities, studies various subjects, and develops a wide variety of interests. Unfortunately, It is a big problem to decide what will be there career.

Profession recommendations should be available for everyone. So without barriers. This project is created to give you an opportunity to answer your question.

This project gives an overview of all important machine learning algorithms and parsing tools that you can use for implementation of your dream

Introduction

Choosing a career is one of the most important decisions you will make in life. It's about so much more than deciding what you will do to make a living. To start with, think about the amount of time we spend at work. We are on the job approximately 71% of every year. Over our lifetimes, this comes to roughly 31½ years out of the 45 years most of us spend working, from the beginning of our careers until retirement. The importance of selecting a career with which we are satisfied cannot be overemphasized.

While some people are lucky enough to just know what they want to do and end up in satisfying careers without giving it much thought, most of us are not. Many people don't put enough effort into choosing occupations or pick them for the wrong reasons. Maybe they choose careers that seem secure or pay well. They then end up unhappy. The best way to make sure that doesn't happen to you is to make a well-thought out decision. Our application will help you! So how to build it? What kind of tools we use?

Aim of research

1. What kind of applications and frameworks we need for research?
2. How to parse data for prediction?
3. Which Algorithm we use for prediction of your project?
4. Building Web Application.

Background/Literature Review

Tools for research

Setting up your Python environment for Machine Learning can be a tricky task. If you've never set up something like that before, you might spend hours fiddling with different commands trying to get the thing to work. But we just want to get right to the ML!

1) ***Set up Python 3 and Pip***

The first step is to install pip , a Python package manager:

sudo apt-get install python3-pip

Using pip, we'll be able to install any Python package that's indexed in the Python Package Index with a simple pip install your_package . You'll see soon how we use it to set up our virtual environment too.

Next, we'll set Python 3 to be the default when running either the pip or python commands from command line. This makes using Python 3 easier and more convenient. If we didn't do this, then if we wanted to use Python 3, we'd have to remember to type out pip3 and python3 every time!

2) ***Set up Anaconda***

The open-source Anaconda Distribution is the easiest way to perform Python/R data science and machine learning on Linux, Windows, and Mac OS X. With over 15 million users worldwide, it is the industry standard for developing, testing, and training on a single machine, enabling individual data scientists

Download from webpage:

https://www.anaconda.com/distribution/

3) ***Install libraries***

pip is the package installer for Python. You can use pip to install packages from the Python Package Index and other indexes.

Please take a look at our documentation for how to install and use pip:

https://pip.pypa.io/en/stable/

For our project we need to install:

Pip install bs4

Pip install request

Pip install numpy

Pip install sklearn

Pip install matplotlib

4) ***Install Web2Py***

From : ***http://www.web2py.com/init/default/download***

5) ***Connection to Oracle database***

Pip install cx_Oracle

Methods and Materials

1)Machine Learning

List of Common Machine Learning Algorithms

- Linear Regression.
- Logistic Regression.
- Decision Tree.
- SVM.
- Naive Bayes.
- KNN.
- K-Means.
- Random Forest.

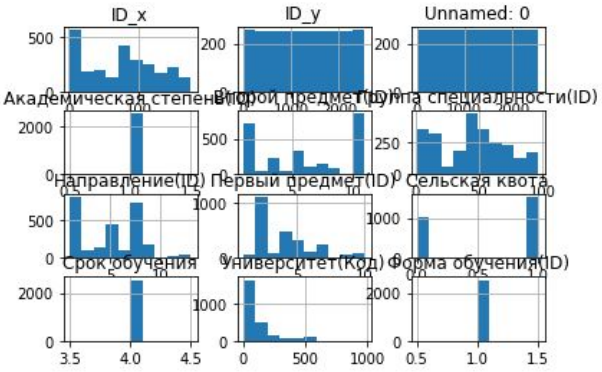
For our algorithm we are decided to use Logistic Regression and KNN

```
df_iris.head()
```

amed: 0	ID_x	Название(RU)	Название(KK)	Описание(RU)	Описание(KK)	Специальность(Код)	Направление(ID)	Академи стег
0	1	Дошкольное обучение и воспитание	Мектепке дейінгі оқыту және тәрбиелеу	Объекты профессиональной деятельности \nОбъект...	NaN	5B010100	1	
1	1	Дошкольное обучение и воспитание	Мектепке дейінгі оқыту және тәрбиелеу	Объекты профессиональной деятельности \nОбъект...	NaN	5B010100	1	
2	1	Дошкольное обучение и воспитание	Мектепке дейінгі оқыту және тәрбиелеу	Объекты профессиональной деятельности \nОбъект...	NaN	5B010100	1	
3	1	Дошкольное обучение и воспитание	Мектепке дейінгі оқыту және тәрбиелеу	Объекты профессиональной деятельности \nОбъект...	NaN	5B010100	1	
4	1	Дошкольное обучение и воспитание	Мектепке дейінгі оқыту және тәрбиелеу	Объекты профессиональной деятельности \nОбъект...	NaN	5B010100	1	

Visualization of professions

```
df_iris.hist()
plt.show()
```



Test and train

```
#split the data into training and test data randomly
iris_train, iris_test = train_test_split(df_iris, test_size = 0.3, random_state = 10)
```

```
print(df_iris.shape)
print(iris_train.shape)
print(iris_test.shape)
```

```
(2550, 17)
(1785, 17)
(765, 17)
```

```
iris_train["Специальность(Код)"].unique().tolist()
```

Predictions of new data:

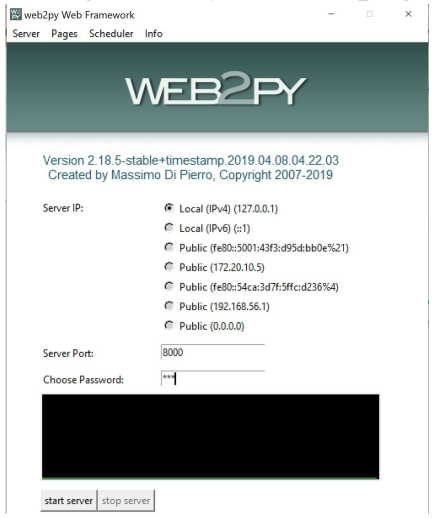
Let's predict the value for one set of inputs

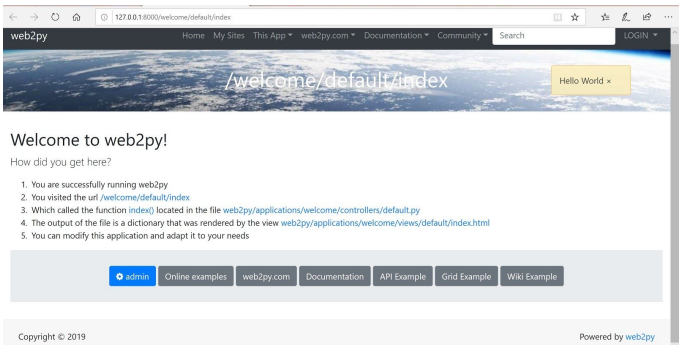
```
]: l = [1,10,1,1]
   a = np.array([l])
   testing_prob = pd.DataFrame(columns=unique_targets)

]: for elem in unique_targets:
       lr = models[elem]
       ls = lr.predict_proba(a)
       testing_prob[elem] = ls[:,1]

]: testing_prob.idxmax(axis = 1)[0]
```

*2) **Web2Py Web Framework***

Web2Py-Free open source full-stack framework for rapid development of fast, scalable, secure and portable database-driven web-based applications.
So, in order to implement our web-site we are using python based Web2Py framework.
Main ***features*** of the Web2py:

- Cross-platform framework that provides support for Windows, Unix/Linux, Mac, Google App Engine, and many other platforms.
- Built-in components to handle HTTP requests, HTTP responses, cookies, and sessions as well.
- Ability to read multiple protocols.
- Security to data against all possible threats such as cross-site scripting, injection flaws, and execution of infected files.
- Follows model-view-controller (MVC) pattern.
- Support for role-based access control and internationalization.
- Allows users to embed jQuery for Ajax and UI effects.
- ***It gives us your web page with available port, where you are admin.***





Data and Results

Data for prediction:

First of all to make prediction everyone should make research. We decided to parse data from:

Joo.kz - webpage with dataset of specialties

https://postupi.online/professii/

Headhunter.com

To parse data we used beautiful soup and request forms of libraries.
Joo.kz provided us data of universities. Professions was parsed from https://postupi.online/professii/, Headhunter.com

Implementation:

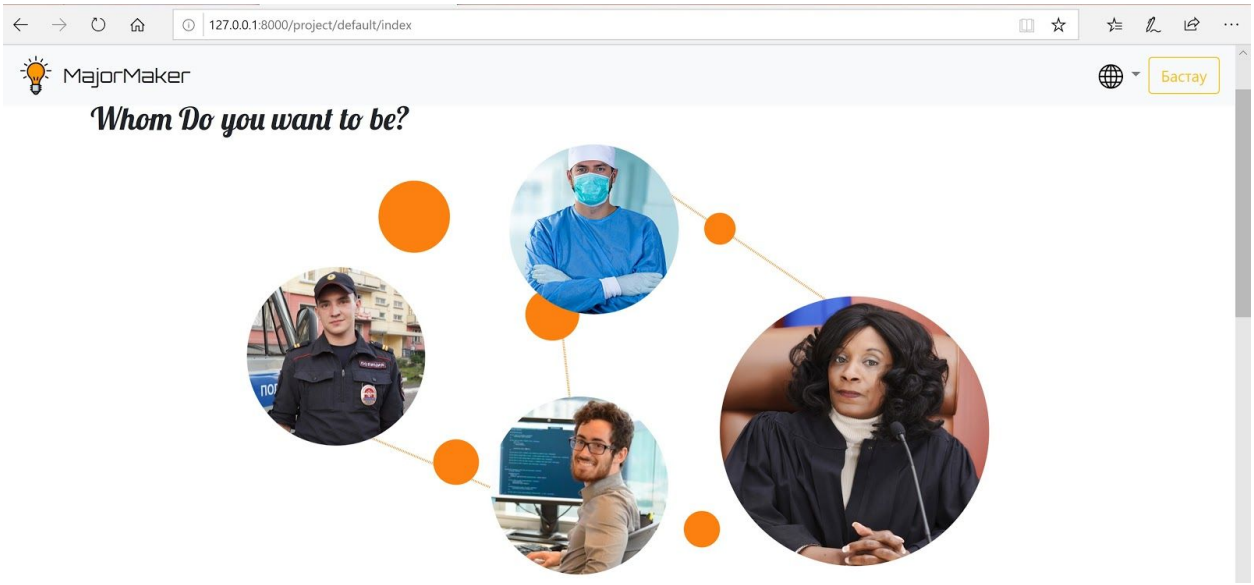
```
import bs4
import requests
```

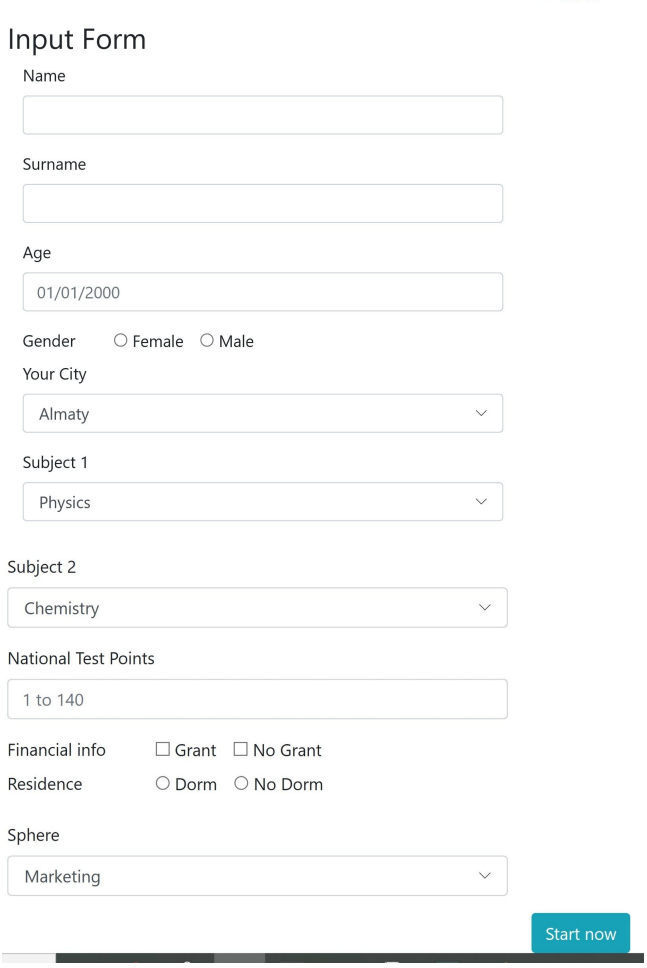
```
page=0
for i in range(37):
    i+=30
    print(i)
    for j in range(25):
        url = 'https://postupi.online/professii/?fletter='+str(i)+'&page_num=' + str(j)
        res =requests.get(url)
        soup = bs4.BeautifulSoup(res.text, 'lxml')
        slovo = soup.find_all("a", class_="flex-panel__h2")
        da = str(slovo).split(',')
        asm.append(da)
        page+=1
        print('Succesfully parsed page: {}'.format(page))
```

```
for i in range(3529):
    if(len(newl[i])>2):
        done = str(newl[i]).split('/">')
        if(len(done)==2):
            da = str(done[1]).split('<')
            if(len(da)==2):
                da = da[0].replace("'", "")
                lol.append(da)
```

Results:

- **Web2Py** uses ***MVC pattern***, therefore our code would be in different files.
- ***View:*** Here is the view of our web page. We used Bootstrap4, Javascript and CSS.





- ***Model:*** Here is the brain of web2py or your backend.
- **Default.py:**

```
def index():
    print request.vars
    if request.vars.submit:
        db.students.insert(name=request.vars.student_name,
            surname=request.vars.student_surname,
            age=request.vars.student_age,
            gender=request.vars.gender,
            city=request.vars.city_name,
            subject1=request.vars.subject1,
            subject2=request.vars.subject2,
            points=request.vars.points,
            grants=request.vars.grants,
            dorm=request.vars.dorm,
            sphere=request.vars.sphere)
        print "Record was inserted successfully"
    return locals()
# ---- API (example) -----
@auth.requires_login()
def api_get_user_email():
    if not request.env.request_method == 'GET': raise HTTP(403)
    return response.json({'status':'success', 'email':auth.user.email})

# ---- Smart Grid (example) -----
@auth.requires_membership('admin') # can only be accessed by members of admin groupd
def grid():
    response.view = 'generic.html' # use a generic view
    tablename = request.args(0)
    if not tablename in db.tables: raise HTTP(403)
    grid = SQLFORM.smartgrid(db[tablename], args=[tablename], deletable=False, editable=False)
    return dict(grid=grid)
```

- ***Controller:*** Here is the connection with database
- **db.py:**

```
db.define_table("students",
    Field("name"),
    Field("surname"),
    Field("age"),
    Field("gender"),
    Field("city"),
    Field("subject1"),
    Field("subject2"),
    Field("points"),
    Field("grants"),
    Field("dorm"),
    Field("sphere")
    )
```

```
db = DAL(configuration.get('db.uri','oracle://system/zhanna2011@test'),
         pool_size=configuration.get('db.pool_size',10),
         migrate_enabled=configuration.get('db.migrate',True),
         check_reserved=['all','oracle'])
```

- After inputting data in the form, our web page will give the recommendation of professions.

Discussion

1) The hardest part of our project was gathering data in other words collecting csv file.
2) Also, it is troublesome to connect to Oracle database
3) Recommender system works pretty well

Conclusion

To sum up, in this project we learned building Web Application applying Machine Learning. To be exact, we learned collecting the dataset by ourselves, by parsing it with python libraries. Then we learned building the website with Python Full-Stack Framework- Web2Py. In addition, we learned building the Recommender System via Machine Learning algorithm.

References

1. Joo.kz
2. https://postupi.online/professii/
3. Headhunter.com