

소프트웨어융합최신기술 과제 #1

20163089 김동언

Data Handling

기상청 데이터 파싱

- 기상청 날씨정보를 조회하는 페이지의 URL을 Capture
- requests 라이브러리를 통해 페이지 크롤링
- BeautifulSoup4 라이브러리를 통해 내용 파싱
- 이 부분을 함수화

The screenshot shows the KMA website's 'past_table.jsp' page. The URL bar highlights the query string: `?stn=108&yy=2012&obs=10`. The Network tab in the browser's developer tools shows the request to `past_table.jsp?stn=108&yy=...` with its headers and query string parameters.

stn: 지점코드
yy: 요청년도
obs_key: 기후항목

기상청 페이지 URL Syntax

월	2월	3월	4월	5월	6월
9.9	52.32	0	0.6	17.3	16.1
0.3	-17.1	5.2	3.5	19.0	19.1
0.8	-14.5	2.7	1.1	17.0	18.1
0.6	-5.2	3.0	2.2	14.3	18.1
0.8	-4.5	3.1	3.1	13.5	19.1
0.6	-3.2	1.8	2.6	13.1	19.1
0.0	-11.5	-0.2	0.6	12.2	18.1
0.5	-12.1	-0.3	4.6	14.2	19.1

페이지 Element Tree

```
<p class="table_topinfo">[ 일최저기온(℃) ] 108
2012년 /p
<table class="table_develop" summary="일평균기온">
  <caption>최저기온</caption>
  <colgroup>...</colgroup>
  <thead>
    <tr>
      <th>...</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td scope="row">2일</td>
      <td>-8.3</td>
      <td>-17.1</td> == $0
      <td>-5.2</td>
      <td>3.5</td>
      <td>19.0</td>
      <td>19.2</td>
      <td>21.7</td>
      <td>26.8</td>
      <td>20.8</td>
    </tr>
  </tbody>
</table>
```

기후 데이터셋 구축

- 앞서 작성한 기상청 데이터 파싱 함수를 통해 데이터를 수집
- 이를 `pandas.DataFrame`으로 변환
- 가공 후 csv로 저장

```
Crawling KMA (yy: 2010, stn: 108)... OK (1.92 sec)
Crawling KMA (yy: 2011, stn: 108)... OK (1.72 sec)
Crawling KMA (yy: 2012, stn: 108)... OK (1.51 sec)
Crawling KMA (yy: 2013, stn: 108)... OK (1.66 sec)
Crawling KMA (yy: 2014, stn: 108)... OK (1.80 sec)
Crawling KMA (yy: 2015, stn: 108)... OK (1.58 sec)
```

	date	average_temp	min_temp	max_temp	rel_humadity	rainfall
2186	2015-12-27	-5.7	-8.9	-1.0	39.0	0.0
2187	2015-12-28	-5.7	-9.5	-1.0	38.4	0.0
2188	2015-12-29	-3.2	-8.7	2.9	47.6	0.0
2189	2015-12-30	0.3	-3.5	4.3	67.6	2.5
2190	2015-12-31	0.7	-1.9	2.8	81.3	0.0


기후 데이터셋 구축 과정

모기 데이터 정리

- 생태포털에서 모기포집수 데이터 다운로드 및 정리
- 관측소별 위,경도 유형 데이터 정리한 `json` 파일 작성

<input type="checkbox"/>	<input type="text" value=".."/>
<input type="checkbox"/>	<input type="text" value="A_2012.csv"/>
<input type="checkbox"/>	<input type="text" value="A_2013.csv"/>
<input type="checkbox"/>	<input type="text" value="A_2014.csv"/>
<input type="checkbox"/>	<input type="text" value="B_2013.csv"/>
<input type="checkbox"/>	<input type="text" value="B_2014.csv"/>
<input type="checkbox"/>	<input type="text" value="C_2013.csv"/>
<input type="checkbox"/>	<input type="text" value="C_2014.csv"/>
<input type="checkbox"/>	<input type="text" value="D_2013.csv"/>
<input type="checkbox"/>	<input type="text" value="D_2014.csv"/>
<input type="checkbox"/>	<input type="text" value="E_2011.csv"/>
<input type="checkbox"/>	<input type="text" value="E_2012.csv"/>
<input type="checkbox"/>	<input type="text" value="E_2013.csv"/>
<input type="checkbox"/>	<input type="text" value="E_2014.csv"/>
<input type="checkbox"/>	<input type="text" value="F_2013.csv"/>
<input type="checkbox"/>	<input type="text" value="F_2014.csv"/>
<input type="checkbox"/>	<input type="text" value="G_2013.csv"/>
<input type="checkbox"/>	<input type="text" value="G_2014.csv"/>
<input type="checkbox"/>	<input type="text" value="H_2014.csv"/>

모기 데이터 다운로드 후 일괄 정리 (shell script 활용)

 jupyter observatory.json ✓ 10/03/2017

File	Edit	View	Language
1	<pre>{ "A": {"type": 2, "lat": 37.5276, "lng": 126.913}, "B": {"type": 1, "lat": 37.5197, "lng": 37.5327, "lng": 126.907}, "D": {"type": 3, "lat": 37.5028, "lng": 126.894}, "E": {"type": 1, "lat": 37.4954, "lng": 126.898}, "G": {"type": 1, "lat": 37.517, "lng": 126.883}, "I": {"type": 1, "lat": 37.494, "lng": 126.905}, "J": {"type": 1, "lat": 37.5114, "lng": 126.922}, "L": {"type": 3, "lat": 37.5273, "lng": 126.889}, "N": {"type": 3, "lat": 37.5233, "lng": 126.881}, "O": {"type": 3, "lat": 37.5254, "lng": 126.933}, "Q": {"type": 1, "lat": 37.527, "lng": 126.922}, "R": {"type": 1, "lat": 37.5263, "lng": 126.895}, "T": {"type": 1, "lat": 37.5236, "lng": 126.916} }</pre>		

관측소 정보 Json 작성

모기 데이터셋 구축

- 앞서 정리한 데이터 파일과 `json` 파일을 기준으로 데이터를 `pandas.DataFrame` 으로 변환
- 가공 후 csv로 저장

```

Building U2013... OK (0.00 sec)
Building U2014... OK (0.00 sec)

```

	date	mosquito_count	level	obs_label	obs_type	latitude	longitude
10920	2014-10-31	6.0	1	U	2	37.5236	126.916
10921	2014-11-01	14.0	2	U	2	37.5236	126.916
10922	2014-11-02	13.0	2	U	2	37.5236	126.916
10923	2014-11-03	1.0	1	U	2	37.5236	126.916
10924	2014-11-04	2.0	1	U	2	37.5236	126.916

모기 데이터셋 구축 과정

데이터셋 구축

- 앞서 구축한 모기 데이터셋과 기후 데이터셋을 날짜를 기준으로 병합한다.
- `column` 중 불필요한 것, Target Data, Feature Data를 분류한다.
- 여기까지의 데이터셋을 한번 csv로 저장한다.
- 불필요한 `column`을 제거한다.
- Random Forest Algorithm을 통해 예측 모델을 만든다.

```

Reading cached dataset... OK (0.02 sec)
Adding KMA[average_temp]... OK (23.23 sec)
Adding KMA[min_temp]... OK (19.69 sec)
Adding KMA[max_temp]... OK (19.79 sec)
Adding KMA[rel_humidity]... OK (19.61 sec)
Adding KMA[rainfall]... OK (19.68 sec)

```

	obs_type	latitude	longitude	average_temp	average_temp_past	min_temp	min_temp_past	max_temp	max_temp_past	rel_humidity	rel_humid
10920	2	37.5236	126.916	15.0	14.2	12.8	7.7	18.2	20.8	62.1	
10921	2	37.5236	126.916	17.3	15.0	15.1	12.8	22.9	18.2	63.4	
10922	2	37.5236	126.916	12.0	17.3	7.1	15.1	15.4	22.9	50.5	
10923	2	37.5236	126.916	9.3	12.0	3.8	7.1	14.8	15.4	40.9	
10924	2	37.5236	126.916	11.3	9.3	5.4	3.8	18.5	14.8	60.5	

데이터셋 병합 과정

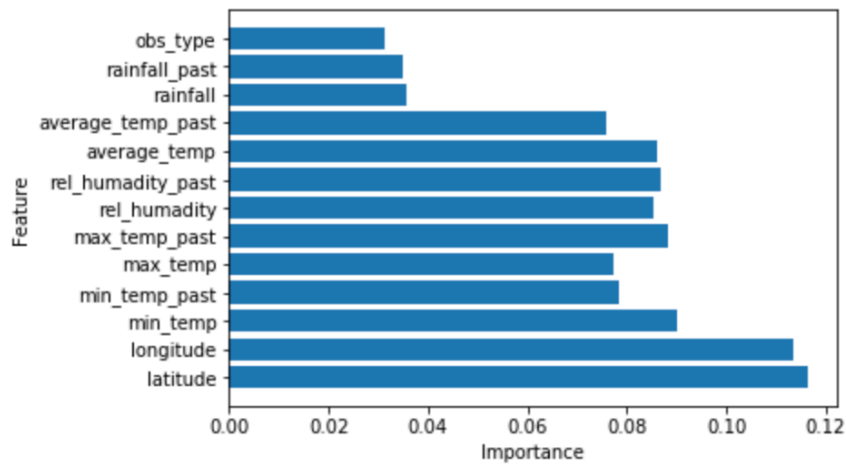
	0	1	2	3	4	5	6	7	8	9	...	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277
actual	1	1	5	4	6	5	3	6	1	1	...	3	4	3	2	1	3	5	1	1	2
predict	1	1	6	4	8	6	1	7	1	6	...	4	4	4	2	1	1	1	1	1	1

2 rows × 3278 columns

Train Accuracy :: 0.982869098993

Test Accuracy :: 0.46644295302

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                        max_depth=None, max_features='auto', max_leaf_nodes=None,
                        min_impurity_decrease=0.0, min_impurity_split=None,
                        min_samples_leaf=1, min_samples_split=2,
                        min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,
                        oob_score=False, random_state=None, verbose=0,
                        warm_start=False)
```



Hyper Parameter 를 기본값으로 두고 실행했을 때 (정확도 46%)

Parameter 최적화

- RandomForestClassifier 의 Hyper Parameter를 조정해가며 최적의 지점을 탐색한다.
- Feature Importance 에 기반해 Feature 를 조정해본다.

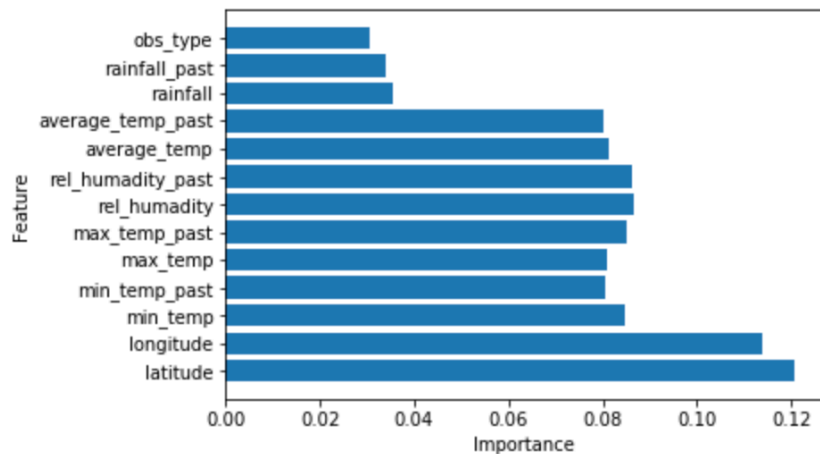
	0	1	2	3	4	5	6	7	8	9	...	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277
actual	1	1	5	4	6	5	3	6	1	1	...	3	4	3	2	1	3	5	1	1	2
predict	1	1	6	4	8	6	1	7	1	1	...	3	4	4	2	1	1	3	1	1	1

2 rows x 3278 columns

Train Accuracy :: 1.0

Test Accuracy :: 0.512202562538

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                        max_depth=26, max_features='auto', max_leaf_nodes=None,
                        min_impurity_decrease=0.0, min_impurity_split=None,
                        min_samples_leaf=1, min_samples_split=2,
                        min_weight_fraction_leaf=0.0, n_estimators=500, n_jobs=-1,
                        oob_score=False, random_state=9, verbose=0, warm_start=False)
```



tree의 수를 늘리고, overfitting을 방지하기 위해 `max_depth` 를 제한 (정확도 51%)

모기 포집수 예측

- 2017년 기후 데이터를 크롤링 하고, 모기 포집수를 앞서 만든 예측 모델을 통해 예측한다.

```
# load trained model
with open("trained_model", "rb") as dumpfile:
    trained_model = pickle.load(dumpfile)

# Prediction
start_time = time.time()
print("Predicting...", end='\t')
predict_df = pd.DataFrame({'date': get_date_keys_monthly(TARGET_YEAR, TARGET
print("OK (%.2f sec)" % (time.time() - start_time))
display(predict_df)

# Save result
predict_df.to_csv("predict_result.csv")
```

```
Crawling kma data...    OK (1.56 sec)
Reformatting data...   OK (0.00 sec)
Predicting...          OK (0.21 sec)
```

	date	level
0	2017-08-01	3
1	2017-08-02	1
2	2017-08-03	1
3	2017-08-04	3
4	2017-08-05	3
5	2017-08-06	1
6	2017-08-07	1
7	2017-08-08	3

예측 과정

최종 제출

- Jupyter Notebook 파일 `mq_analysis_code.ipynb` (코드)
- 가공한 데이터셋 파일 `build_dataset`
- 예측 모델 `predict/trained_model`
- 예측 결과 `predict/predict_result.csv`
- 라이브 코딩 영상 `live_coding.mp4`