Week11

Created By	DongGu Kim
Last Edited	@May 29, 2020 12:57 AM
Property	
Tags	

내부 공개용 [데이터 유출문제 때문에 깃허브 업로드용은 문자메시지 삭제한 버전임]

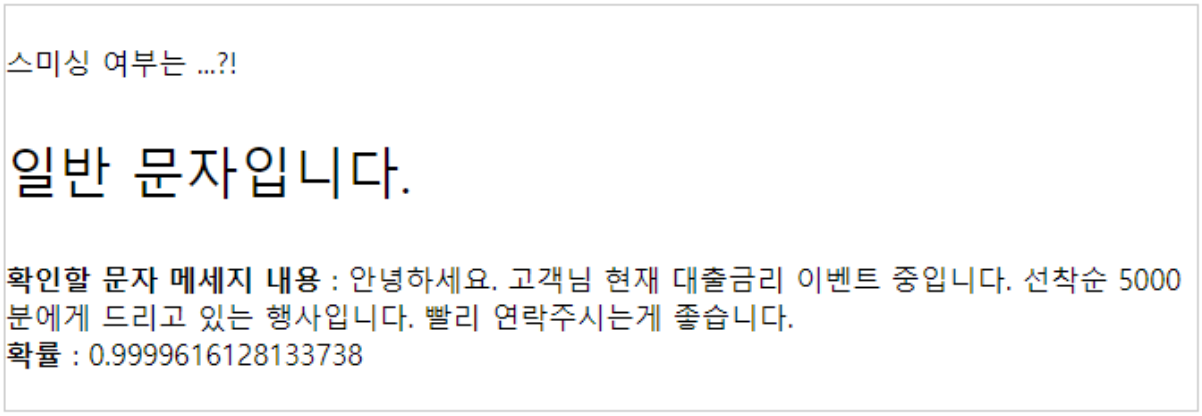
1. SVM 모델 Deploy 및 웹 개발 완료

2. SVM 및 LGBM Negative Record 확인

---

1. SVM 모델 Deploy 및 웹 개발 완료

- Flask를 활용한 '스미싱 문자 예측' 사이트 기본 기능 구현 완료 [SVM 모델]



- apps.py

```
import flask
import pickle
import pandas as pd

from sklearn.svm import LinearSVC
from sklearn.calibration import CalibratedClassifierCV
from scipy import sparse
import joblib

from flask import Flask, jsonify, request
import os

######################################################
############# INPUT DATA 전처리 ##################
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer

vectorizer=TfidfVectorizer(ngram_range=(1, 3),
    min_df=2,
    max_features=10000,
    sublinear_tf=True,
    lowercase=False,
    use_idf=True)


model = joblib.load('model/svm_model.pkl')


app = flask.Flask(__name__, template_folder='templates')

@app.route('/', methods=['GET', 'POST'])
def main():
```

```
    if flask.request.method == 'GET':
        # Just redner the initial form, to get input
        return(flask.render_template('main.html'))

    if flask.request.method == 'POST':
        # Extract the input
        message = flask.request.form['message']
        loaded_vectorizer = pickle.load(open('model/test_vec.pickle', 'rb'))
        message_data = loaded_vectorizer.transform([message])
        prediction = model.predict(message_data)
        proba = model.predict_proba(message_data)

        if (prediction == 1):
            prediction = "스미싱 문자"
            proba = proba[0][1]
        else:
            prediction = "일반 문자"
            proba = proba[0][0]

        return flask.render_template('main.html',
                                     original_input={'확인할 문자 메세지 내용':message,
                                                     '확률':proba},
                                     result=prediction,
                                     )

if __name__ == '__main__':
    app.run()
```

- main.html

```
<!doctype html>
<html>
<style>
form {
    margin: auto;
    width: 35%;
}

.result {
    margin: auto;
    width: 35%;
    border: 1px solid #ccc;
}

</style>

<head>
    <title>Smishing? Smash!</title>
</head>
<form action="{{ url_for('main') }}" method="POST">
    <fieldset>
        <legend>문자메시지 내용 입력:</legend>
        Message:
        <input name="message" type="text" required>
        <br>
```

```
        <br>
        <input type="submit">
    </fieldset>
</form>

<br>
<div class="result" algin="center">
    {% if result %}
    <br> 스미싱 여부는 ...?!
    <p style="font-size:30px">{{ result }}입니다.</p>
        {% for variable, value in original_input.items() %}
            <b>{{ variable }}</b> : {{ value }} <br>
      {% endfor %}
    <br>
    {% endif %}
</div>

</html>
```

- LIME을 이용한 '스미싱 문자 메시지 판단 근거' 제시 기능 구현 예정

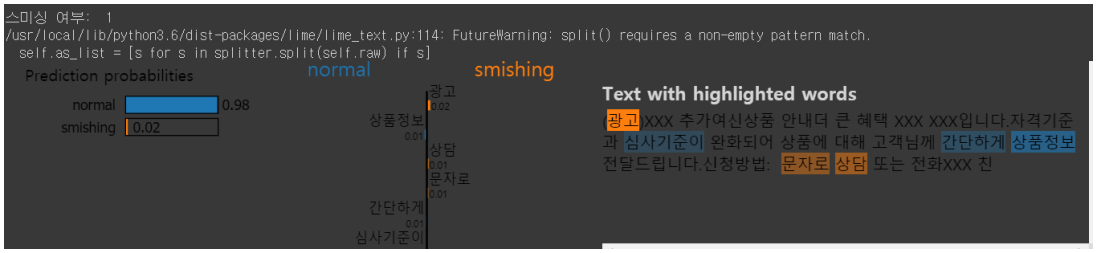
2. SVM 및 LGBM Negative Record 확인

1. SVM 30개 모델에서 전부 틀린 Record Primary Key 확인
    - 틀린 갯수: 80개
        - false positive 1개 (실제로 스미싱 문자인데, 예측결과 일반문자)
        - false negative 79개 (실제로 일반 문자인데, 예측결과 스미싱문자)

    - False Record 특징 (false negative에 초점을 맞춤)
        - LGBM 모델에 비해 각 클래스일 '확률'이 5:5인 경우가 많음.
        - 대책: '확률'이 5:5에 가까운 경우 다른 방법을 통해 예측확률 재산정

2. LGBM 30개 모델에서 전부 틀린 Record data Primary Key 확인
    - 틀린 갯수: 212개
        - false positive 6개 (실제로 스미싱 문자인데, 예측결과 일반문자)
        - false negative 196개 (실제로 일반 문자인데, 예측결과 스미싱문자)

    - False Record 특징(false negative에 초점을 맞춤)
        - SVM 모델에 비해 '확률'이 극단적인 경우가 많음.
        - LGBM의 '연속 앙상블(이전 모델의 오답에 대한 가중치가 후속 모델에 반영됨)'의 영향일 것이란 추측 ('False Record'의 꼬리를 계속 물다보니 예측 확률이 극단적으로 산출되는 것 같음)



3. SVM 및 LGBM 공통으로 틀린 Record
    - 총 38개 (전부 false negative)