머신러닝을 활용한 제주노지감귤 비상품량 예측

**Forcasting of Jeju orange by machine learning**

2019312048 유새하

김효정

이관구

이상협

홍형근

**Abstract(이관구)**

머신러닝을 활용한 제주노지감귤 출하량(->비상품량) 예측의 필요성 설명- ex) 버려지는 감귤을 최소화하고 수확시기와 최적판매시기를 예측해볼 수 있었다.

- 수학적 모델 회귀분석을 사용했다.

- 분석 모델은 RNN 사용해서 분석해보았다.

**ⅰ. 서론(개념설명)**

1. 감귤의 종류와 현재 연구의 필요성(날씨데이터와 연관)- 김효정
2. 감귤의 재배에 영향을 줄 수 있는 날씨요인 (강수량이 얼마이상일때, 일조량이 얼마 이하일 때, 온도가 얼마 이상일때) – 이관구
3. 제주도의 날씨- 김효정

1. 머신러닝의 개념
2. 지도학습(버터, 형근)
3. 회귀분석
4. KNN

k-nearest neighbor는 데이터를 분류하고 새로운 데이터 포인트의 카테고리를 결정할 때 K 개의 가장 가까운 포인트를 선점하고 그중 가장 많이 선택된 포인트의 카테고리로 이 새로운 데이터를 분류하는 방법이다. k-nearest neighbor에서 고려해야 할 사항은 알고리즘의 핵심 부분이 대상 포인트와의 거리에 대한 측정이고, 이를 계산하는 방법으로 무조건 유클리드 거리 측정 방식을 사용하는 것을 자제해야 한다. 모든 데이터 열을 이처럼 같은 방식으로 처리하면 생각하지 못한 변수에 의해 오류가 생길 수 있으므로 거리의 제곱을 합산하기 전 각 카테고리에 대한 평균 거리를 빼고 계산하는 방식과 같은 다양한 거리 계산 알고리즘에 대한 논의가 필요하다. 예를 들어 실수 데이터의 경우 유클리드 거리 측정 방식을 사용하고, 범주형 혹은 이진 데이터와 같은 유형의 데이터는 해밍 거리 측정 방식을 사용한다.

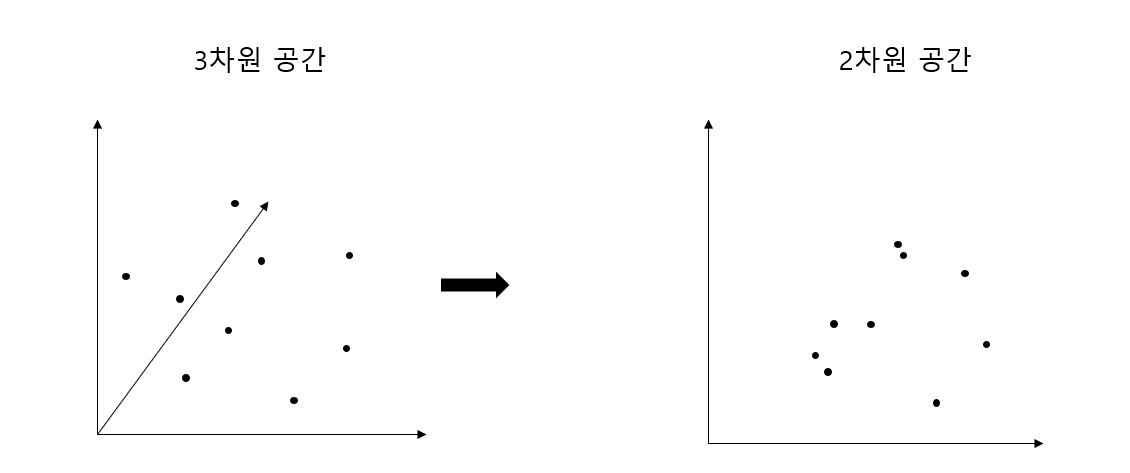
1. SVM

서포트 벡터 머신(support vector machine, SVM[[1]](https://ko.wikipedia.org/wiki/%EC%84%9C%ED%8F%AC%ED%8A%B8_%EB%B2%A1%ED%84%B0_%EB%A8%B8%EC%8B%A0#cite_note-CorinnaCortes-1).[[2]](https://ko.wikipedia.org/wiki/%EC%84%9C%ED%8F%AC%ED%8A%B8_%EB%B2%A1%ED%84%B0_%EB%A8%B8%EC%8B%A0#cite_note-2))은 [기계 학습](https://ko.wikipedia.org/wiki/%EA%B8%B0%EA%B3%84_%ED%95%99%EC%8A%B5)의 분야 중 하나로 패턴 인식, 자료 분석을 위한 [지도 학습](https://ko.wikipedia.org/wiki/%EC%A7%80%EB%8F%84_%ED%95%99%EC%8A%B5" \o "지도 학습) 모델이며, 주로 [분류](https://ko.wikipedia.org/wiki/%EB%B6%84%EB%A5%98)와 [회귀 분석](https://ko.wikipedia.org/wiki/%ED%9A%8C%EA%B7%80_%EB%B6%84%EC%84%9D)을 위해 사용한다. 두 카테고리 중 어느 하나에 속한 데이터의 집합이 주어졌을 때, SVM 알고리즘은 주어진 데이터 집합을 바탕으로 하여 새로운 데이터가 어느 카테고리에 속할지 판단하는 비[확률적](https://ko.wikipedia.org/wiki/%ED%99%95%EB%A5%A0) 이진 [선형 분류](https://ko.wikipedia.org/wiki/%EC%84%A0%ED%98%95_%EB%B6%84%EB%A5%98) 모델을 만든다. 만들어진 분류 모델은 데이터가 사상된 공간에서 경계로 표현되는데 SVM 알고리즘은 그 중 가장 큰 폭을 가진 경계를 찾는 알고리즘이다. SVM은 선형 분류와 더불어 비선형 분류에서도 사용될 수 있다. 비선형 분류를 하기 위해서 주어진 데이터를 고차원 특징 공간으로 사상하는 작업이 필요한데, 이를 효율적으로 하기 위해 [커널 트릭](https://ko.wikipedia.org/w/index.php?title=%EC%BB%A4%EB%84%90_%ED%8A%B8%EB%A6%AD&action=edit&redlink=1)을 사용하기도 한다.

1. RNN
2. 비지도학습(유새하)
3. 차원축소(PCA, Factor analysis)

대부분 실무에서 분석하는 데이터는 매우 많은 특성(feature)들을 가지고 있다. 이러한 데이터를 가지고 머신러닝 알고리즘을 적용해 문제를 해결하려고 한다면, 데이터의 차원이 크기 때문에 학습 속도가 느릴 뿐만아니라 성능 또한 좋지 않을 가능성이 크다. 따라서 데이터의 차원을 줄여 학습속도와 성능을 올려야 한다. 차원축소 방법에는 투영 (projection)과 매니폴드 학습(manifold learning)이 있으며 대표적인 차원 축소 알고리즘인 주성분분석(PCA)이 있다.

* 투영 (projection)

일반적으로 대부분의 실제 데이터셋에서는 모든 데이터의 특성, 즉 차원이 고르게 분포되어 있지 않다. 학습 데이터셋은 고차원 공간에서 저차원 **부분 공간(subspace)**에 위치하게 된다. 즉, 고차원의 데이터의 특성 중 일부 특성으로 데이터를 표현할 수 있다는 말이 된다. 

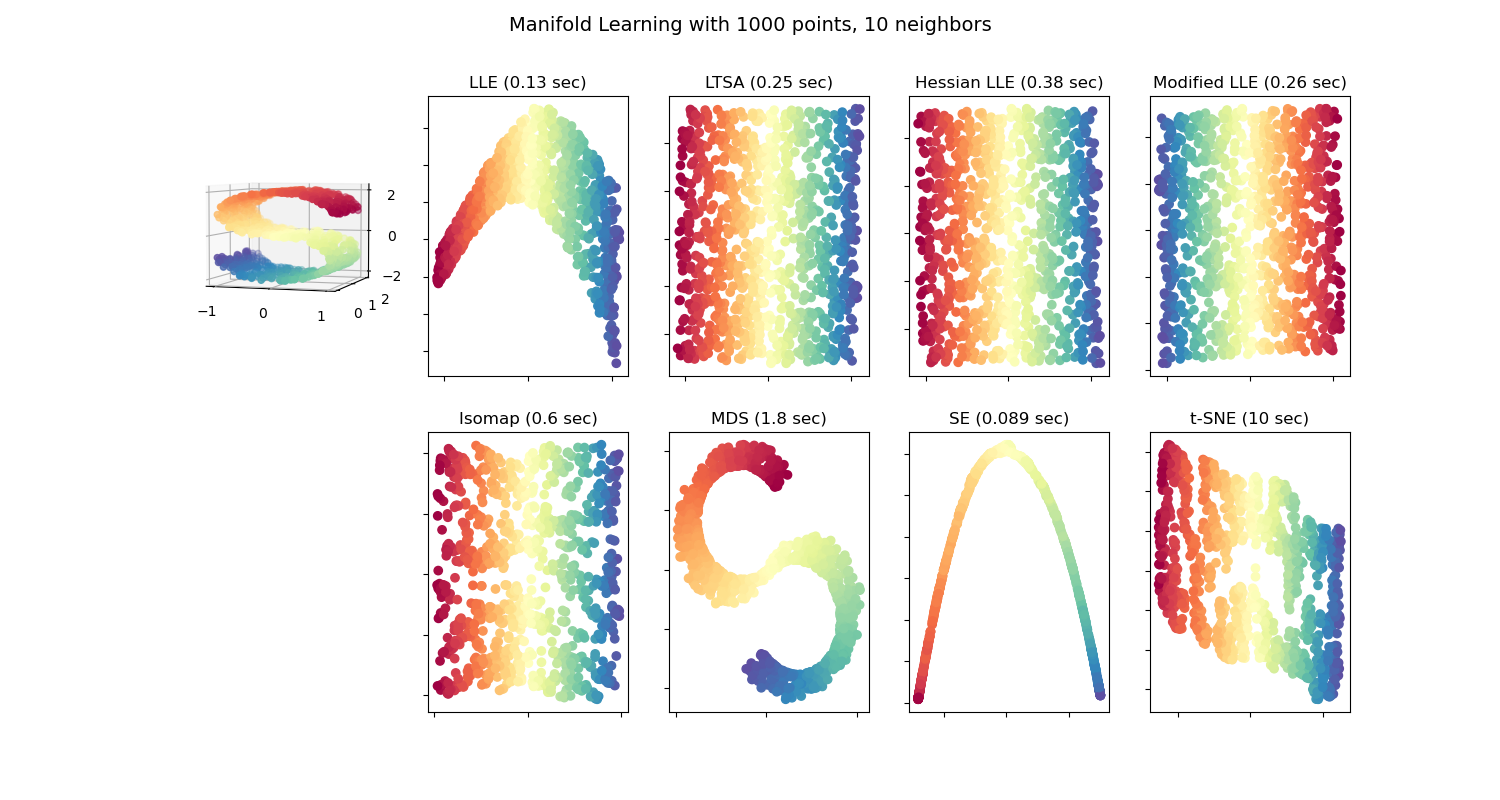
<그림 1>

* 매니폴드 학습(manifole learning)

매니폴드는 다양체라고도 하며 국소적으로 유클리드 공간과 닮은 위상 공간이다. 즉, 국소적으로는 유클리드 공간과 구별할 수 없으나 , 대역적으로 독특한 위상수학적 구조를 가질 수 있다(출처: 위키피디아 ).

대부분의 차원 축소 알고리즘이 이러한 **매니폴드**를 모델링하는 방식으로 동작하며, 이를 **매니폴드 학습(Manifold Learning)** 이라고 한다. 매니폴드 학습은 **매니폴드 가정**(manifold assumption) 또는 **매니폴드 가설**(manifold hypothesis)에 의해, 고차원인 실제 데이터셋이 더 낮은 저차원 매니폴드에 가깝게 놓여 있다고 가정한다.

매니폴드 가정은 종종 다른 가정과 함께 쓰이기도 한다. 예를들어, 분류나 회귀같은 작업을 하기전에 학습 데이터셋을 저차원의 매니폴드 공간으로 표현하면 더 간단하게 문제를 해결할 수 있다라는 가정을 할 수 있다.



<그림 2 <https://scikit-learn.org/stable/modules/manifold.html>>

* 주성분분석(PCA)

주성분 분석은 고차원의 데이터를 저차원의 데이터로 환원시키는 기법을 말한다.

이 때 서로 연관 가능성이 있는 고차원 공간의 표본들을 선형 연관성이 없는 저차원 공간(주성분)의 표본으로 변환하기 위해 [직교 변환](https://ko.wikipedia.org/w/index.php?title=%EC%A7%81%EA%B5%90_%EB%B3%80%ED%99%98&action=edit&redlink=1)을 사용한다. 데이터를 한개의 축으로 사상시켰을 때 그 [분산](https://ko.wikipedia.org/wiki/%EB%B6%84%EC%82%B0)이 가장 커지는 축을 첫 번째 주성분, 두 번째로 커지는 축을 두 번째 주성분으로 놓이도록 새로운 좌표계로 데이터를 [선형 변환](https://ko.wikipedia.org/wiki/%EC%84%A0%ED%98%95_%EB%B3%80%ED%99%98)한다. 이와 같이 표본의 차이를 가장 잘 나타내는 성분들로 분해함으로써 데이터 분석에 여러가지 이점을 제공한다. 이 변환은 첫째 주성분이 가장 큰 분산을 가지고, 이후의 주성분들은 이전의 주성분들과 직교한다는 제약 아래에 가장 큰 분산을 갖고 있다는 식으로 정의되어있다. 중요한 성분들은 [공분산 행렬](https://ko.wikipedia.org/w/index.php?title=%EA%B3%B5%EB%B6%84%EC%82%B0_%ED%96%89%EB%A0%AC&action=edit&redlink=1)의 고유 벡터이기 때문에 직교하게 된다.

주성분분석의 순서는 다음과 같다.

|  |  |
| --- | --- |
| 1 | 공분산 행렬 계산하기 |
| 2 | 고유분해를 통해 고유값과 고유 벡터 계산하기 |
| 3 | 고유값이 큰 순서대로 나열하기 |
| 4 | 지정된 최소 분산 크기 이상을 설명하도록 n번째 고유 벡터까지 선택하기 |
| 5 | 기존 데이터에 선택된 고유벡터를 내적 |

* 인자분석 (Factor analysis)

인자 분석(factor analysis) 또는 요인 분석은 [인자](https://ko.wikipedia.org/wiki/%EC%9D%B8%EC%9E%90)(factor) 또는 [요인](https://ko.wikipedia.org/wiki/%EC%9A%94%EC%9D%B8)이라고 불리는 잠재적으로 적은 숫자의 관찰되지 않은 [변수](https://ko.wikipedia.org/wiki/%EB%B3%80%EC%88%98)(variable)들로, 관찰된 서로 상관인 변수(variable)들 사이에서의 [분산](https://ko.wikipedia.org/wiki/%EB%B6%84%EC%82%B0)(variance)을 설명하기 위한 [통계학](https://ko.wikipedia.org/wiki/%ED%86%B5%EA%B3%84%ED%95%99)적 방법이다. 예를 들어, 6개의 관측된 [변인](https://ko.wikipedia.org/wiki/%EB%B3%80%EC%9D%B8)(variable)들의 분산(variation)은 2개의 관측되지 않은 근본적인 변인을 반영할 수도 있다. 인자 분석은 관측되지 않은 [잠재 변수](https://ko.wikipedia.org/wiki/%EC%9E%A0%EC%9E%AC_%EB%B3%80%EC%88%98)에 대하여 이런 연결된 분산(joint variation)을 찾는다. 관측된 변수들은 "[오차](https://ko.wikipedia.org/wiki/%EC%98%A4%EC%B0%A8)"(error)를 추가하여, 가능한 인자들의 [선형 결합](https://ko.wikipedia.org/wiki/%EC%84%A0%ED%98%95_%EA%B2%B0%ED%95%A9)으로 모형화된다. [요인 분석](https://ko.wikipedia.org/wiki/%EC%9A%94%EC%9D%B8_%EB%B6%84%EC%84%9D)은 독립된 내재 변인을 찾는 것을 목적으로 한다.

1. 군집(clustering) 찾아보니까 clustering은 군집이고 분류는 classification이라고 해서 군집으로 수정했어. 비지도학습은 군집인거같은데 맞아?

군집(clustering)은 분류(Classification)과 혼동된다. 군집과 분류의 차이는 군집은 ‘근접’을 주요요소로, 분류는 ‘포함’을 주요요소로 갖는다는 것이다.

군집은 대상 데이터를 일부 카테고리로 그룹화하는데, 이 때 카테고리에 얼마나 ‘근접’하냐를 근거로 군집을 만든다. 보통 비지도 학습 (Unsupervised learning)에서 활용된다. 비지도학습에서 많이 사용된다.

분류도 마찬가지로 대상 데이터를 일부 카테고리로 그룹화한다. 이 때 카테고리에 ‘포함’되는가를 근거로 분류한다. 카테고리에 포함되지 않는 데이터라면, 새로운 라벨을 보유하게 된다. 지도학습에서 사용한다.

예를 들어 문자를 판별하여, 스팸 보관함으로 분류하는것과 같은 단일분류와 , 수능 점수가 몇 등급에 해당하는지 판별하는 종류의 다중분류가 있다. 다중분류는 비지도학습의 Clustering과 비슷하지만, 가장 큰 차이점은 Category의 도메인이 정의되있는가 그렇지 않은가이다.

지도학습의 분류는 이미 정해진 카테고리(레이블) 안에서 학습하여 새로운 데이터를 분류하지만, 비지도학습의 군집은 정해지지 않은 카테고리(레이블)를 원하는 만큼 생성하여, 군집하는것이 가장 큰 차이점이다.

|  |  |
| --- | --- |
|  |  |
| 군집 (clustering) | 분류 (classification) |

* 의사결정 트리

가장 단순한 classifier 중 하나로, decision tree와 같은 도구를 활용하여 모델을 그래프로 그리는 매우 단순한 구조로 되어 있다. 이 방식은 root에서부터 적절한 node를 선택하면서 진행하다가 최종 결정을 내리게 되는 model이다. 이 트리의 장점은 누구나 쉽게 이해할 수 있고, 결과를 해석할수있다. 예를 들어 yes를 선택했던 것을 no로 바꾸기만 하면 간단하게 로직을 바꿀 수 있다. 가지고 있는 데이터의 Feature를 분석해서 Tree를  Build하는 과정이 제일 중요하다.

* Random Forest

Decision tree가 여러개 모여 Forest를 이룬 것이다. Decision tree보다 작은 Tree가 여러개 모이게 되어, 모든 트리의 결과들을 합하여 더많은 값을 최종결과로 본다.

* Nave Bayes

나이브 베이즈는 확률을 사용한다. 분류기를 만들 수 있는 간단한 기술로써 단일 알고리즘을 통한 훈련이 아닌 일반적인 원칙에 근거한 여러 알고리즘들을 이용하여 훈련된다. 모든 나이브 베이즈 분류기는 공통적으로 모든 특성 값은 서로 독립임을 가정한다. 예를 들어, 특정 과일을 사과로 분류 가능하게 하는 특성들 (둥글다, 빨갛다, 지름 10cm)은 나이브 베이즈 분류기에서 특성들 사이에서 발생할 수 있는 연관성이 없음을 가정하고 각각의 특성들이 특정 과일이 사과일 확률에 독립적으로 기여 하는 것으로 간주한다.

1. 영상/ 이미지 인식

예시를 쓰면 되는건가 영상/이미지 인식 이거 뭐넣을지 다시 확인할게

1. 분석모델링 방법 설명(RNN)- 홍형근

**ⅱ. 실험방법(분석 tool 설명+ )**

**https://www.tensorflow.org/api\_docs/python/tf/keras/preprocessing/sequence/make\_sampling\_table**

1. 분석 tool 비교설명 후(유새하)

1) anaconda (Jupiter notebook)

Jupiter notebook은 IPython처럼 인터엑티브 형식의 라이브 코드를 제공하는 웹 베이스 어플리케이션이다. Jupiter notebook을 이용해 웹 브라우저 안에서 코드를 실행하고, 이를 웹 브라우저에서 확인 할 수 있다. Jupiter notebook은 절차대로 실행하면서 분석 결과를 확인 할 수 있어 딥러닝에서 많이 사용된다. Jupiter notebook을 사용하기 위해서는 아나콘다로 파이썬을 설치해야한다. 이후 위에서 설명한 대로 웹브라우저에서 코딩을 하면 된다.

2) pandas

Pandas는 관계형 또는 레이블이 된 데이터를 쉽고 직관적으로 작업 할 수 있도록 설계된빠르고유연한 데이터 구조를 제공하는 파이썬 패키지이다. 즉 엑셀이나 csv 파일 등을 쉽게 불러와 dataframe으로 변환해 데이터를 다루기 위한 라이브러리이다. 이번 연구에서 Csv 파일을 불러오기 위해 사용했으며, pandas로 불러온 데이터를 보고, 출력할 때도 사용했다. 또한 불러온 데이터로 연구를 할 때에도 사용했다.

연구할 때 사용한 함수 설명 추가) 개발 끝나고 나서

3) numpy

Numpy는 파이썬에서 배열을 쉽게 처리하고 사용할 수 있도록 하는 파이썬 패키지이다. 데이터 분석 프로젝트에서 자주 사용한다. 데이터는 이미지, 오디오, 스트링 등 다양한 형태로 존재하는데, 이들을 컴퓨터가 이해 할 수 있도록 숫자로 변환한다. 이 과정에서, 효율적으로 숫자 데이터를 처리해야 하는데 이 때 numpy를 이용하면 숫자 배열을 이용해 데이터 처리가 용이해진다. 파이썬에 있는 list 자료구조에 비해 빠르고, 메모리를 효율적으로 사용한다. 또한 반복 문 없이도 내장되어 있는 함수를 이용하여 배열에 있는 데이터를 쉽고 간편하게 처리 할 수 있다.

4) tensor flow 사용하는데 비교되는 특징(왜 사용하게 됐는지 장점 설명)

tensorflow 함수 설명(유새하)

텐서플로(TensorFlow)는 다양한 작업에대해 [데이터 흐름](https://ko.wikipedia.org/wiki/%EB%8D%B0%EC%9D%B4%ED%84%B0_%ED%9D%90%EB%A6%84) 프로그래밍을 위한 오픈소스 소프트웨어 라이브러리이다. 심볼릭 수학 라이브러리이자, [인공 신경망](https://ko.wikipedia.org/wiki/%EC%9D%B8%EA%B3%B5_%EC%8B%A0%EA%B2%BD%EB%A7%9D)같은 [기계 학습](https://ko.wikipedia.org/wiki/%EA%B8%B0%EA%B3%84_%ED%95%99%EC%8A%B5) 응용프로그램에도 사용된다. 이것은 구글내 연구와 제품개발을 위한 목적으로 구글 브레인팀이 만들었고 2015년 11월 9일 [아파치 2.0 오픈 소스 라이선스](https://ko.wikipedia.org/wiki/%EC%95%84%ED%8C%8C%EC%B9%98_%EB%9D%BC%EC%9D%B4%EC%84%A0%EC%8A%A4)로 공개되었다.

텐서플로는 텐서 보드를 통해 파라미터의 양상이나 구조도를 그려주어 tensor 사이의 연결관계를 쉽게 알 수 있다. 또한 구글에서 공식적으로 릴리즈하여 전문적이고 안전하다. 따라서 tensorflow를 사용하게 되었다.

# 이번 연구에서는 RNN 모델을 구현하기 위에 tensor flow를 사용할 것이다. 텐서플로우로 RNN을 구현할 때 두가지 단계로 나뉘는데, 첫번째는 cell을 만드는 것이다. Tf.contrib.rnn.BasicRNNCell함수로 만들 수 있는데, 해당 함수는 b에서 설명한다. 두번째 단계는 만든 cell을 실제로 구동하여 입력에 대한 출력값을 뽑아내는 것이다. 이 때 사용할 함수는 tf.nn.dynamic\_rnn이다. 해당 함수는 c에서 설명한다.

# tf.keras.preprocessing.sequence.make\_sampling\_table

# 단어 순위 기반 확률론적 샘플링 테이블을 생성한다. 형태는 다음과 같다.

# Tf.keras.preprocessing.sequence.make\_sampling\_table ( size, sampling\_factor=1e-05 )

# 매개변수

|  |  |  |
| --- | --- | --- |
| size | Int | 샘플링 할 수 있는 단어 수 |
| sampling\_factor |  | Word2vec의 공식 샘플링 계수 |

반환

i번째 항목이 순위 i의 단어가 샘플링 되어야 할 확률인 길이 size의 1차원 numpy배열

skipgrams 의 sampling\_table 인수 를 생성하는 데 된다. sampling\_table[i] 는 데이터 세트에서 i 번째로 가장 많이 사용되는 단어를 샘플링 할 확률이다.

샘플링 확률은 word2vec에 사용 된 샘플링 분포에 따라 생성된다.

# tf.contrib.rnn.BasicRNNCell

cell을 만드는 함수이며 cell이 어떤 형태인지 지정 할 수 있다. 형식은 아래와 같다.

tf.compat.v1.nn.rnn\_cell.BasicRNNCell(  
    num\_units, activation=None, reuse=None, name=None, dtype=None, \*\*kwargs  
)

|  |  |  |
| --- | --- | --- |
| num\_units | int | RNN cell의 단위 수 |
| activation |  | 사용할 비선형성. 기본값은 tanh이다. |
| reuse | bool | 기존 범위에서 변수를 재사용할지 여부 |
| name | string | 레이어의 이름 |
| dtype |  | 레이어의 기본 dtype |
| \*\*kwargs |  | 셀을 구성할 때 공통 계층 속성에 대한 dict. 키워드 이름 속성 |

# tf.nn.dynamic\_rnn

# RNNCell로 지정된 순환 신경망을 만든다. 형식은 아래와 같다.

tf.compat.v1.nn.dynamic\_rnn(  
    cell, inputs, sequence\_length=None, initial\_state=None, dtype=None,  
    parallel\_iterations=None, swap\_memory=False, time\_major=False, scope=None  
)

|  |  |
| --- | --- |
| cell | RNCell의 인스턴스 |
| inputs | RNN입력 |
| sequence\_length | 크기가 int32/int64인 벡터. 배치 요소의 시퀀스 길이를 지날 때 상태를 복사하고 출력을 0으로 만드는데 사용된다. 선택사항이므로 입력하지 않아도 된다. |
| initial\_state | RNN의 초기상태. 선택사항 |
| dtype | 초기 상태 및 예상 출력에 대한 테이터의 유형. 선택사항 |
| parallel\_iterations | 병렬로 실행할 반복 횟수이다. 기본값은 32이다. |
| swap\_memory |  |
| time\_major | Bool. True이면 tensor 모양이 [max\_time, batch\_size, depth]로 지정되어 있어야 하며 false이면  [batch\_size, max\_time, depth]로 지정되어 있어야 한다. |
| scope | 생성된 하위 그래프에 대한 variablescope. 기본값은 rnn |

2. 데이터 전처리(standard or minmax 스케일링), train test split

Multicollinearity(다중공산성) - 박사님께 여쭤보기(선대내용- 습도, 강수량 )(Variance Inflation Factor)

* 정 안되면 구글링(이관구)

3. 모델링(선형회귀분석?)

RNN을 활용한 모델링 설명  (변수 값들, 그렇게 설정한 이유) – 같이

**ⅲ. 결과해석 및 고찰()**

회귀모델 사용할 경우 평가지표(MSE,MAE,MAPE 등 어떤 평가지표 사용할 것인지)

유의미한 예측 성능을 얻었는지? (정확도)

여러 번 코드를 짰을 때의 결과값 -> 제일 좋은거 선정-

**코드 짜는건 -> 디도나 카페 집중해서 매 2시간마다 비교하기 (매일, 세미 오프라인)**

**점심먹고 만나서 디도나 카페 가기(매일 저녁먹기 전까지, 내일부터 시작)**

**ⅳ. 결론(Conclusion)**

가장 일반화 성능이 좋은 모델  (21년도나 20년도랑 비교했을때 얼마만큼의 예측 성능을 보였는지 발표)- 정확도는 얼마이고~~~