

# Online Retail dataset 분석 & 모델 생성

2019112087 정보통신공학과 이규영

Python Programming for Data Science

# Contents

- ▶ Dataset 설명
- ▶ Datset 시각화
- ▶ Project 목표
- ▶ Dataset 전처리
- ▶ 모델 생성 및 평가

01

# Dataset 설명

## ▶ UCI: Online Retail II Data Set

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.5	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.4	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.8	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.4	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.4	17850.0	United Kingdom

01

# Dataset 설명

## ▶ UCI: Online Retail II Data Set

변수 이름	변수 유형	설명
InvoiceNo	Object	6자리 정수 거래상품명세서 번호 (C 포함시 환불의미)
StockCode	Object	5자리 제품의 고유 식별 번호
Quantity	Int	각 품목의 거래량
InvoiceDate	Object	거래가 이루어진 날짜와 시간
UnitPrice	Float	제품 1개당 가격
CustomerID	Float	5자리 고객의 고유 식별 번호
Country	Object	거래가 이루어진 국가

02

## Dataset 시각화

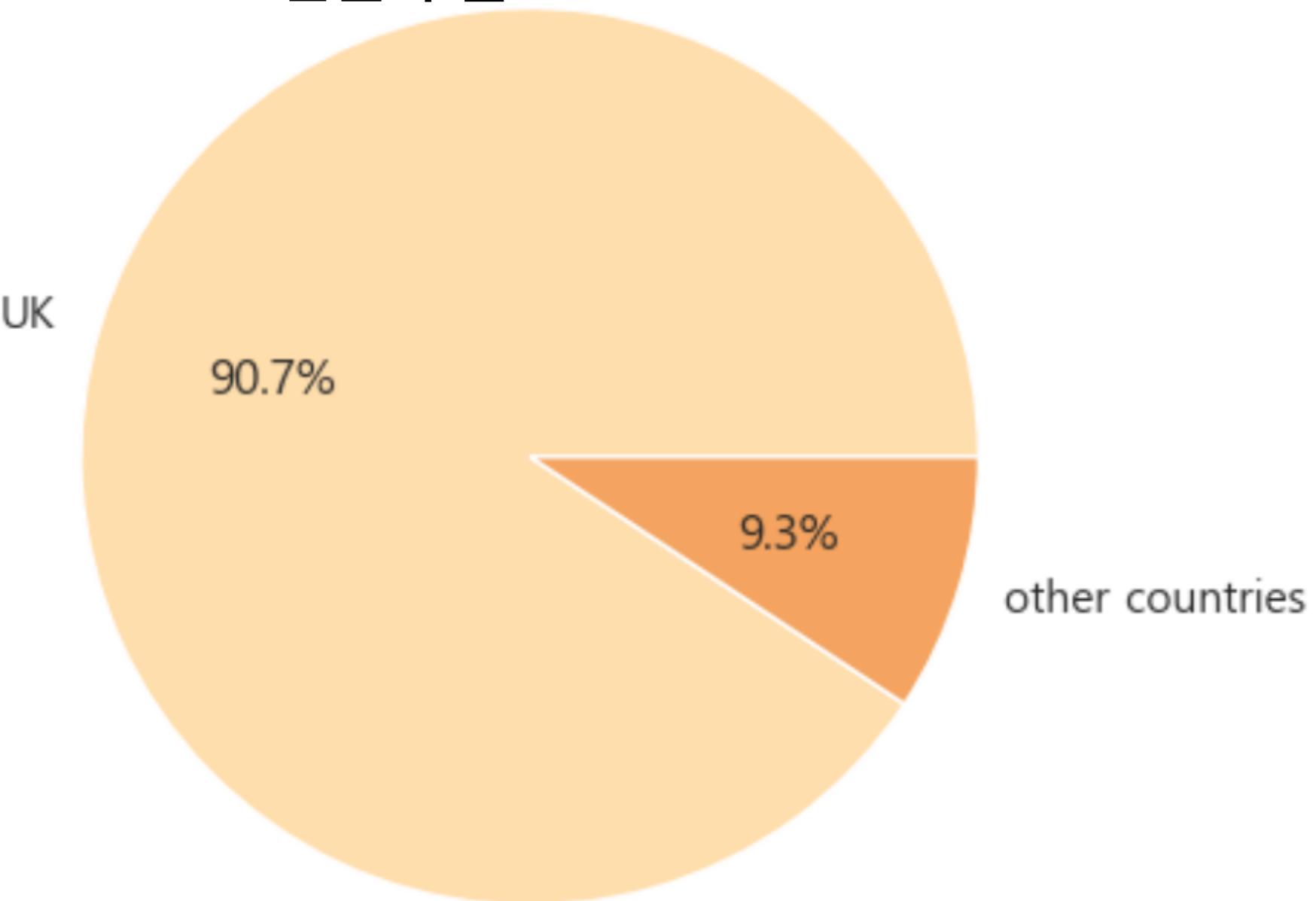
▶ United Kingdom이 대부분을 차지



02

## Dataset 시각화

- ▶ 영국의 online retail dataset임을 추론



02

# Dataset 시각화

## ▶ 월별 판매량 추이



02

# Dataset 시각화

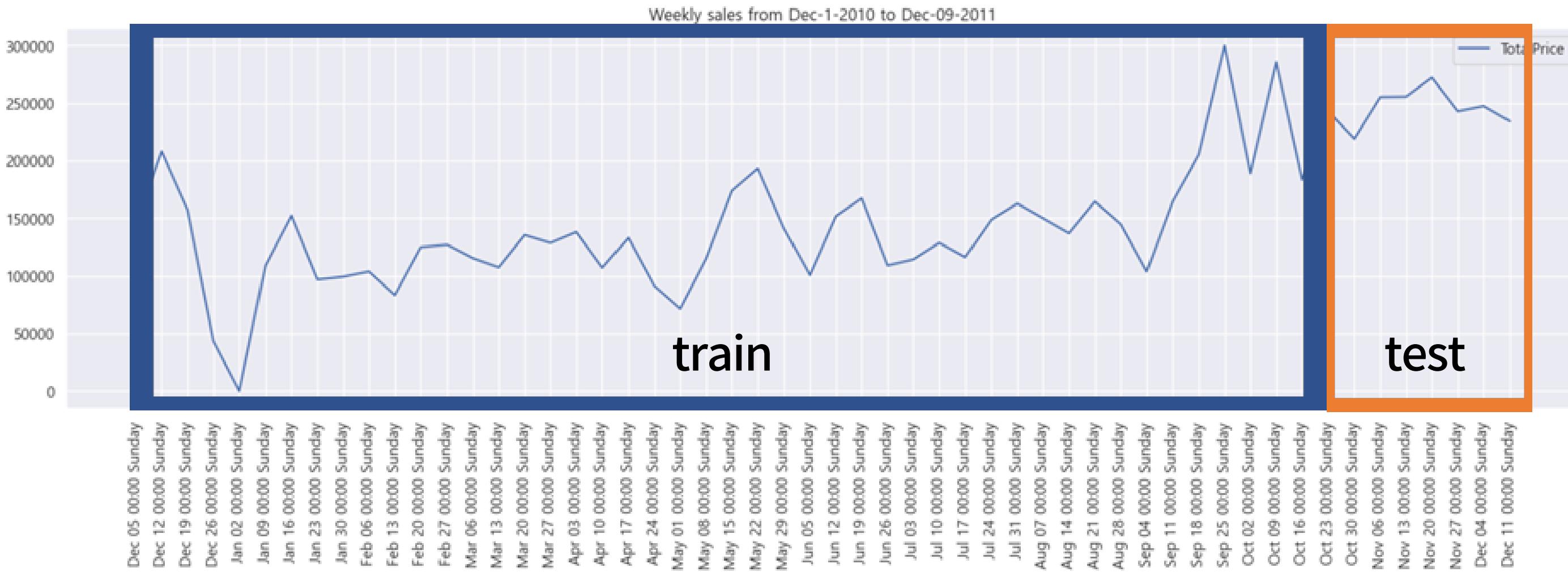
## ▶ 주별 판매량 추이



03

# Project 목표

▶ 2010년 12월부터 2011년 10월까지의 data -> 2011년 11,12월 소비 추이 분포 예측



# Dataset 전처리

▶ InvoiceNo열의 환불처리 case 개수 전체행의 1.7% 차지->삭제하여 구매의 경우로만 모델 생성

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
141	C36379	D	Discount	-1	2010-12-01 09:41:00	27.5	14527.0	United Kingdom
154	C36383	35004C	SET OF 3 COLOURED FLYING DUCKS	-1	2010-12-01 09:49:00	4.7	15311.0	United Kingdom
235	C36391	22556	PLASTERS IN TIN CIRCUS PARADE	-12	2010-12-01 10:24:00	1.6	17548.0	United Kingdom
236	C36391	21984	PACK OF 12 PINK PAISLEY TISSUES	-24	2010-12-01 10:24:00	0.3	17548.0	United Kingdom
237	C36391	21983	PACK OF 12 BLUE PAISLEY TISSUES	-24	2010-12-01 10:24:00	0.3	17548.0	United Kingdom
238	C36391	21980	PACK OF 12 RED RETROSPOT TISSUES	-24	2010-12-01 10:24:00	0.3	17548.0	United Kingdom
239	C36391	21484	CHICK GREY HOT WATER BOTTLE	-12	2010-12-01 10:24:00	3.5	17548.0	United Kingdom

환불 case는  
Quantity<0

InvoiceNo에 환불상품 개수는 9288 개입니다 데이터셋의 1.7139409015166756 % 를 차지합니다.

InvoiceNo에 환불상품 개수는 0 개입니다 데이터셋의 0.0 % 를 차지합니다.

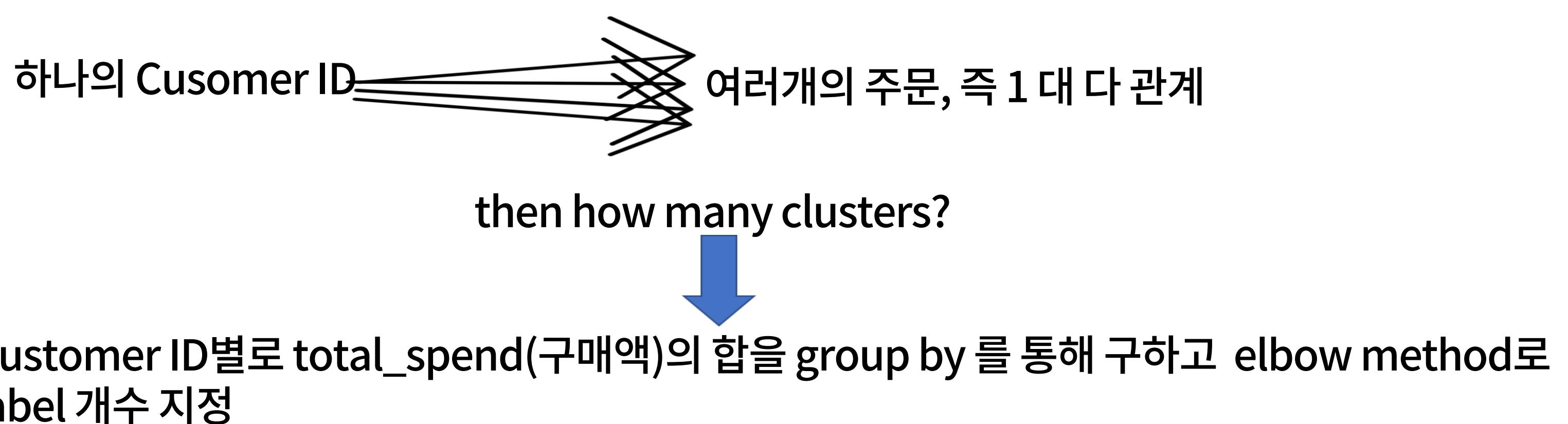
# Dataset 전처리

- ▶ 구입 시간과 날짜가 구매액에 영향을 주는지 파악하기 위해 quarter(분기), month, week 열 추가
- ▶ Quantity x Unit Price 한 total\_spend 열 추가하여 행별 구매액 확인

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	Quarter	Month	Week	total_spend
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.5	17850.0	United Kingdom	4	12	48	15.3
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.4	17850.0	United Kingdom	4	12	48	20.3
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.8	17850.0	United Kingdom	4	12	48	22.0
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.4	17850.0	United Kingdom	4	12	48	20.3
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART	6	2010-12-01 08:26:00	3.4	17850.0	United Kingdom	4	12	48	20.3

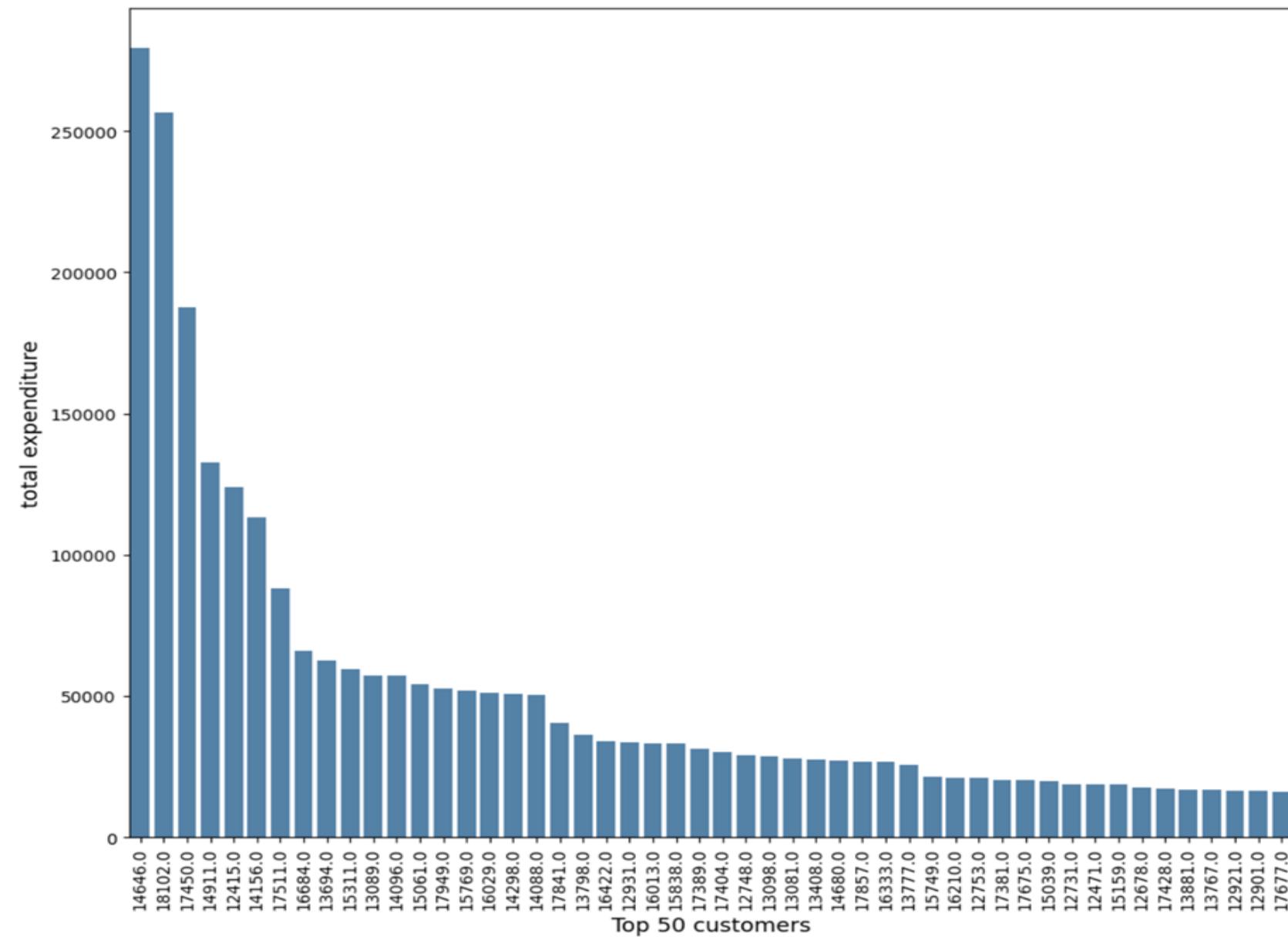
## Dataset 전처리

- ▶ total\_spend의 정확한 값보다 추이를 예측하는 것 -> 2011년 11,12월 spend\_label 예측(target)
- ▶ 그럼 spend\_label을 몇개로 설정해야하나?



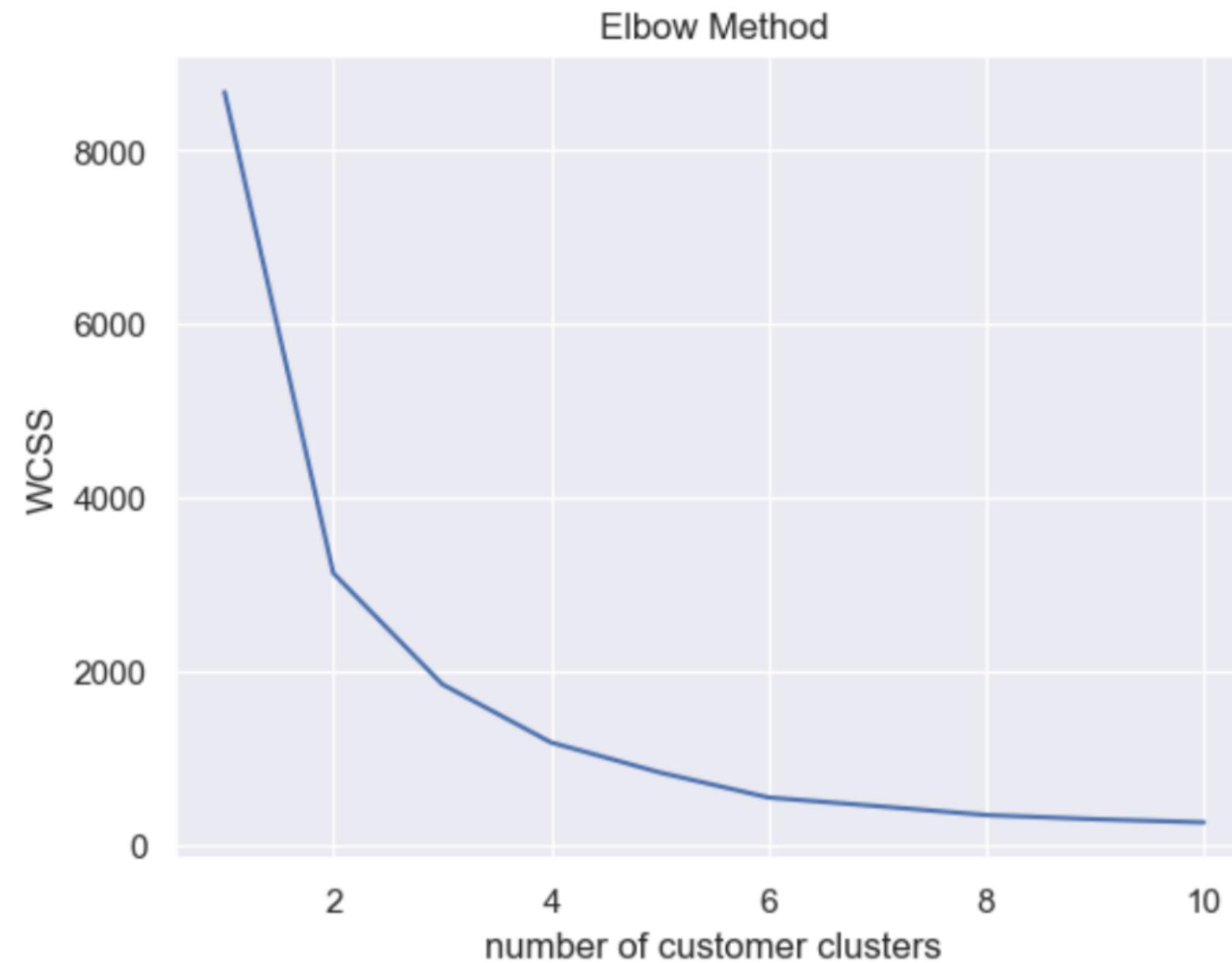
# Dataset 전처리

▶ Customer별로 total\_spend 합계 계산



# Dataset 전처리

- ▶ Elbow Method로 cluster의 개수, 즉 `spend_label`의 개수가 3개가 최적값임을 확인



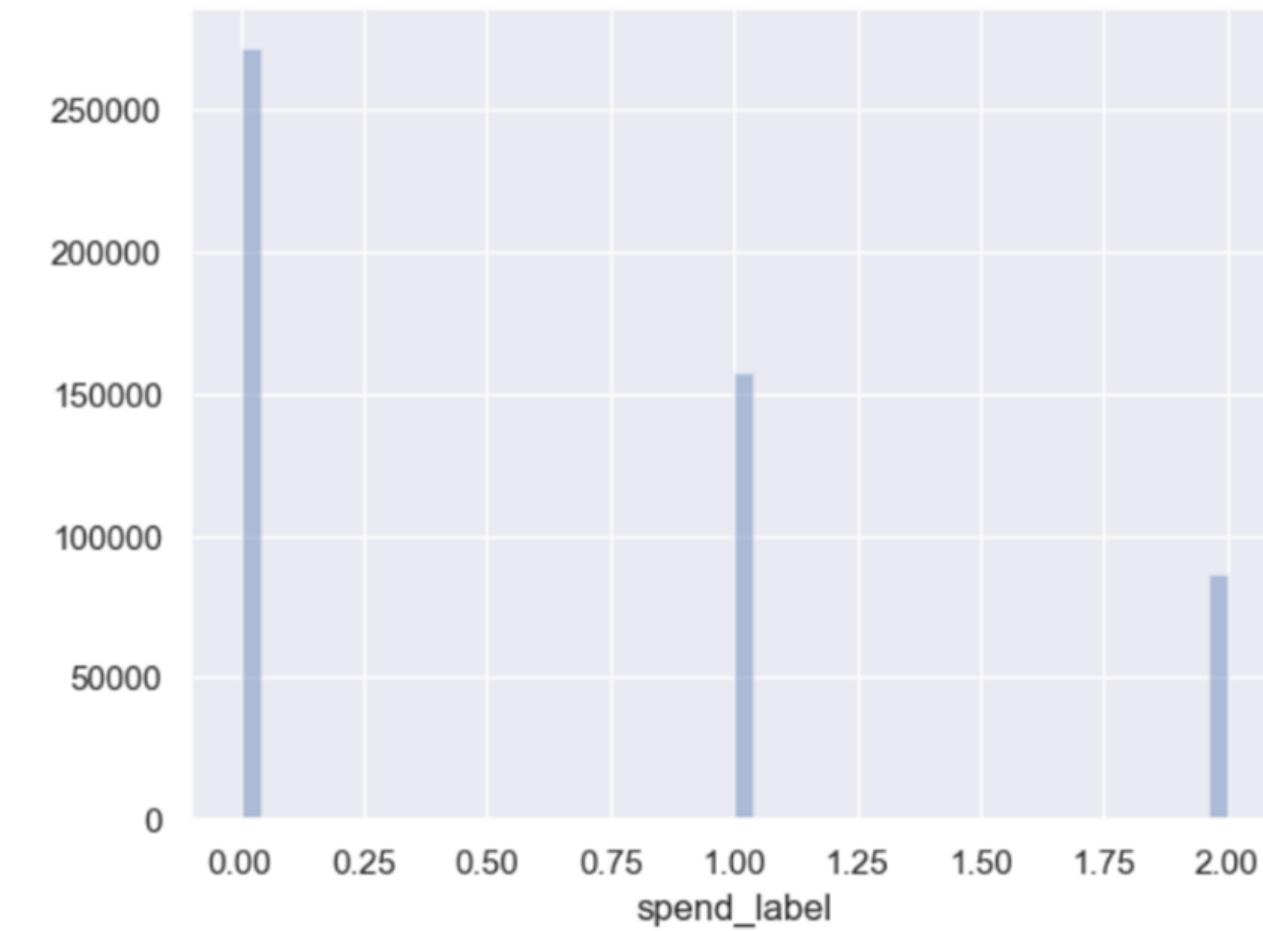
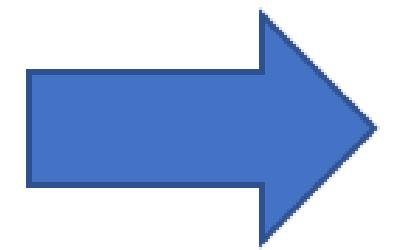
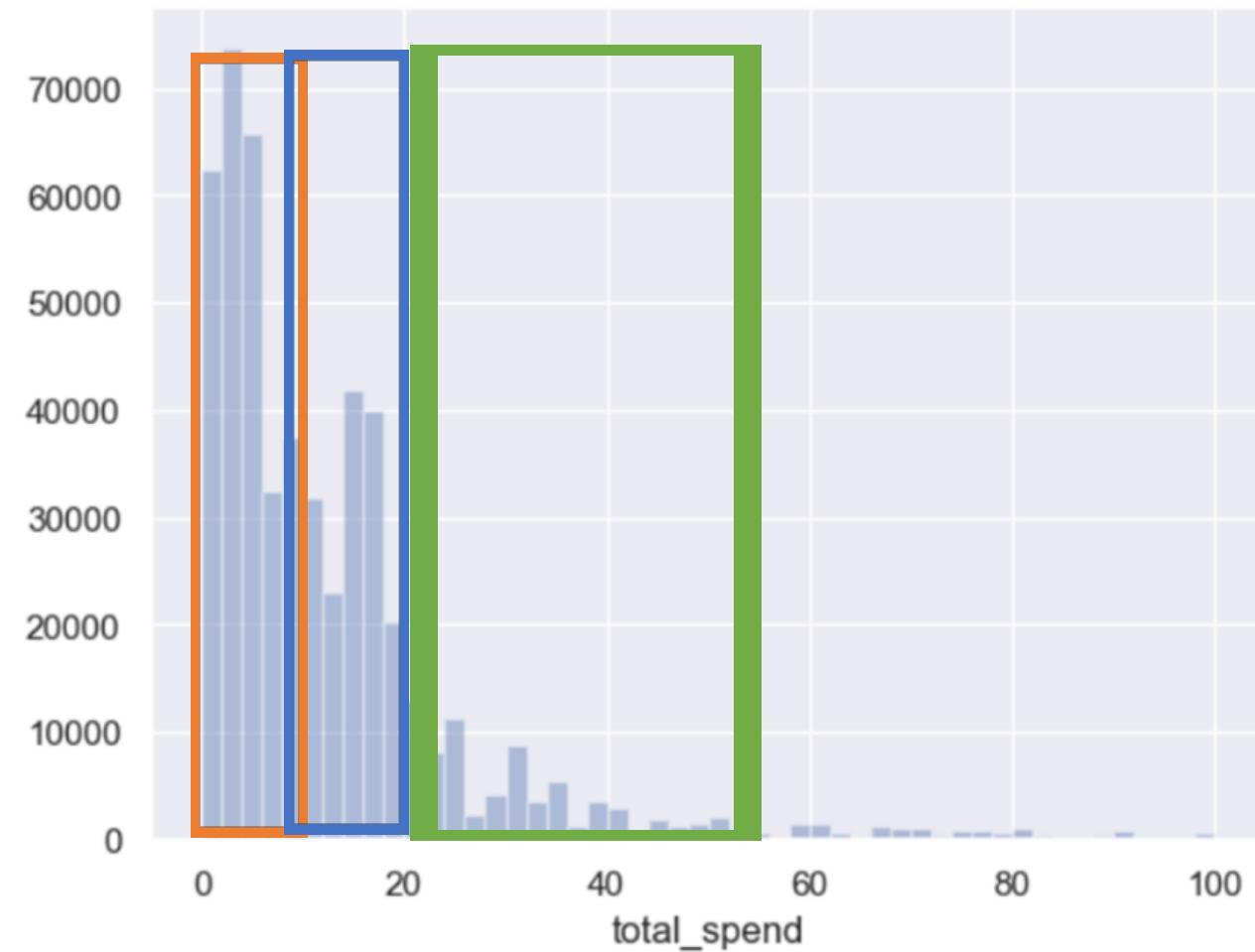
# Dataset 전처리

▶ k-means clustering 적용 -> 3개의 클러스터로 군집화



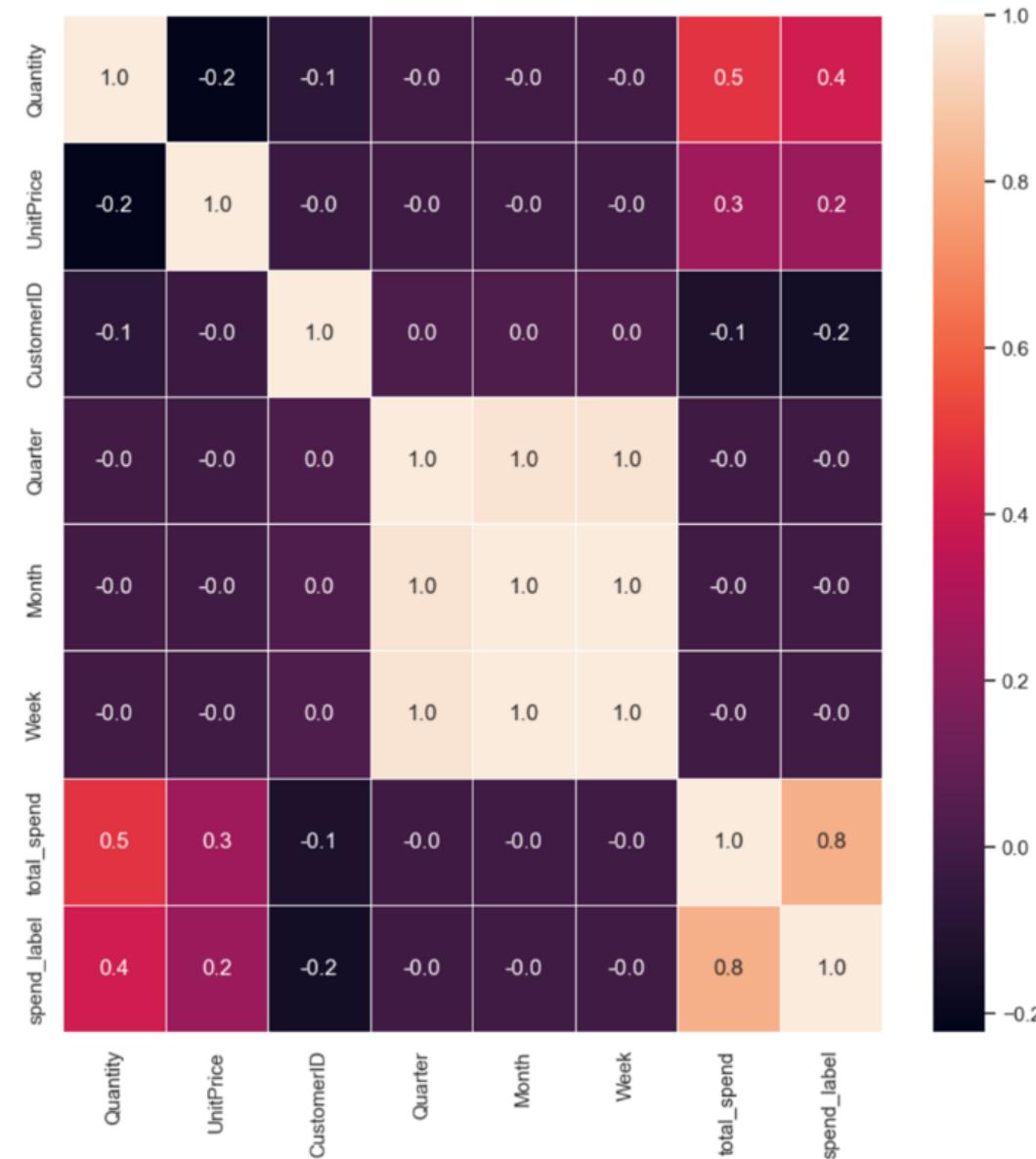
# Dataset 전처리

- ▶ total\_spend의 분포를 3개로 나눈 spend\_label 열 생성

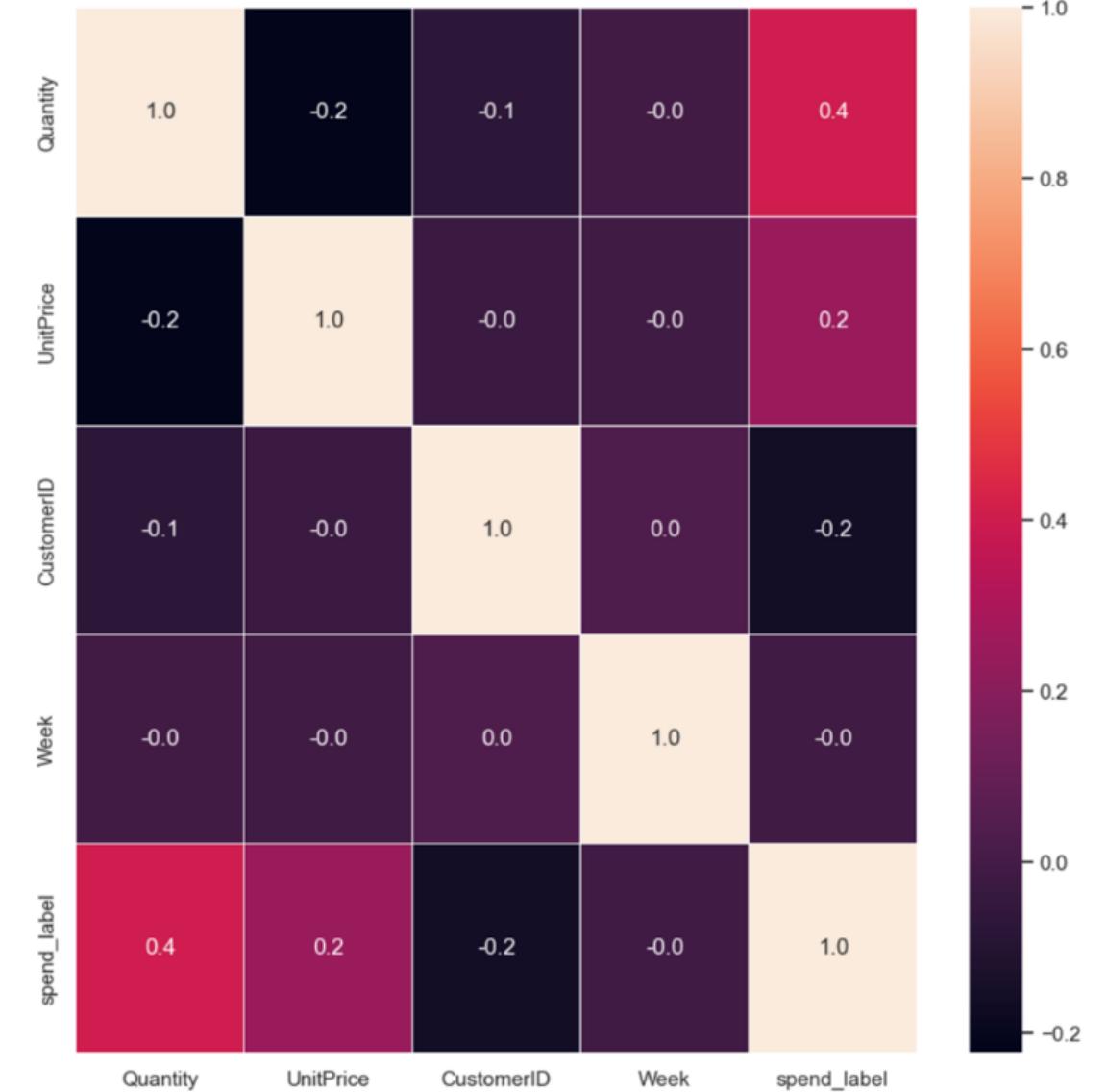


# Dataset 전처리

▶ 머신러닝 모델 적용전 heatmap 확인후 column간 correlation 확인



Quarter, Month,  
total\_spend 열 drop



# Dataset 전처리

- ▶ 2011년 11월 이전의 Quantity, UnitPrice, Week, spend\_label 열을 학습한 모델을 기반으로 2011년 11월 이후의 spend\_label 예측

```
df_train = data3[data3.InvoiceDate < '2011-11-01']
df_test = data3[data3.InvoiceDate >= '2011-11-01']
```

```
x_train = df_train.drop(labels=['InvoiceNo', 'StockCode', 'Description',
                                'Country', 'spend_label', 'InvoiceDate', 'CustomerID'], axis=1)
y_train = df_train.spend_label

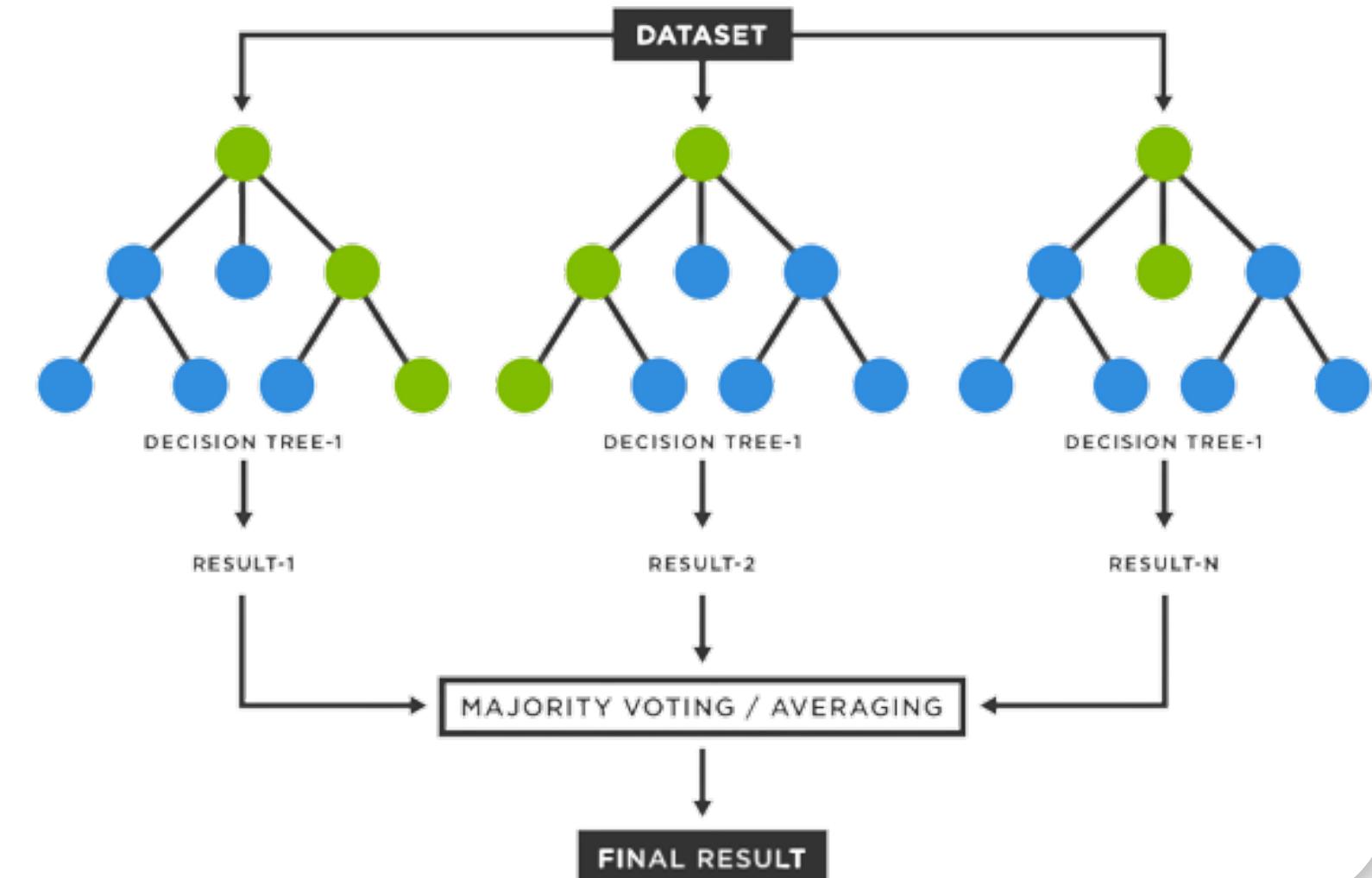
x_test = df_test.drop(labels=['InvoiceNo', 'StockCode', 'Description',
                             'Country', 'spend_label', 'InvoiceDate', 'CustomerID'], axis=1)
y_test = df_test.spend_label
```

- ▶ Description 열은 Unique한 개수가 3885개로 많아 numerical하게 바꾸기에 무리

# Model 생성 및 평가

## ▶ Random Forest 모델 적용 이유

- 여러개의 tree 결합->단일 트리보다 높은 예측 정확도 & 빠른 속도
- tree 개수 증가 -> 과적합 방지
- 각 feature의 중요도 평가 가능



# Model 생성 및 평가

## ▶ Randomized Search CV 이용하여 최적의 hyperparameter 탐색

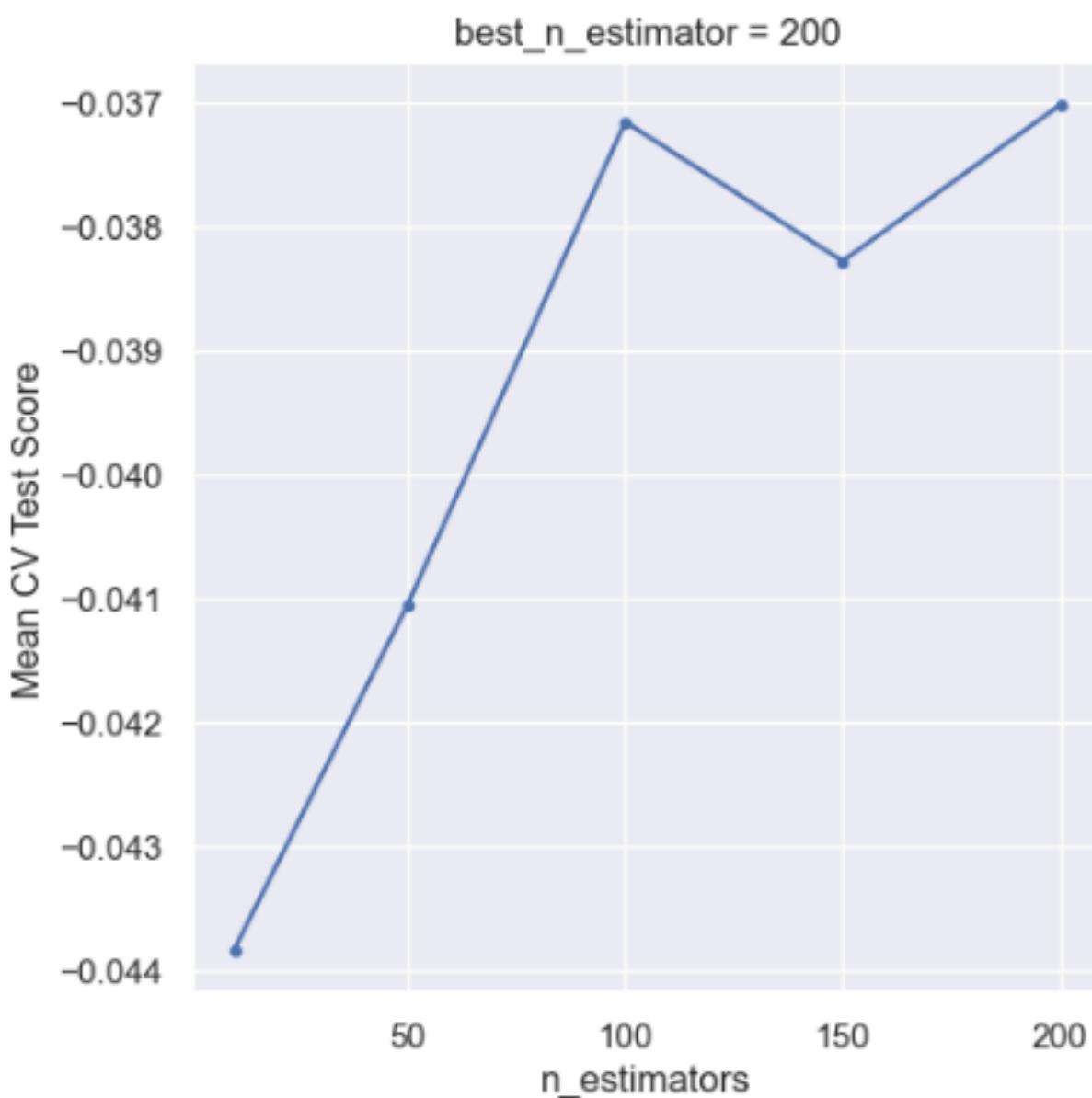
- n\_estimators의 default 값인 100기준 좌우 탐색
- CV=5 적용하여 교차 검증 수행

```
param_distributions = {'n_estimators': [10,50,100,150,200], 'max_features':['auto','sqrt','log2'],
                      'min_samples_leaf':[1,2,5,7,10], 'min_samples_split': [2, 3, 5, 10, 15]}
rf_main = RandomizedSearchCV(RandomForestRegressor(random_state=42), param_distributions, verbose=2,
                             n_iter = 100, cv=5, scoring='rmse_cv', error_score='raise')
%time rf_main.fit(X_train, y_train)
```

Best parameters for Random Forest Regression Model: {'n\_estimators': 200, 'min\_samples\_split': 2, 'min\_samples\_leaf': 1, 'max\_features': 'auto'}

# Model 생성 및 평가

- ▶  $n_{estimator} \rightarrow 200$  이 최적의 값

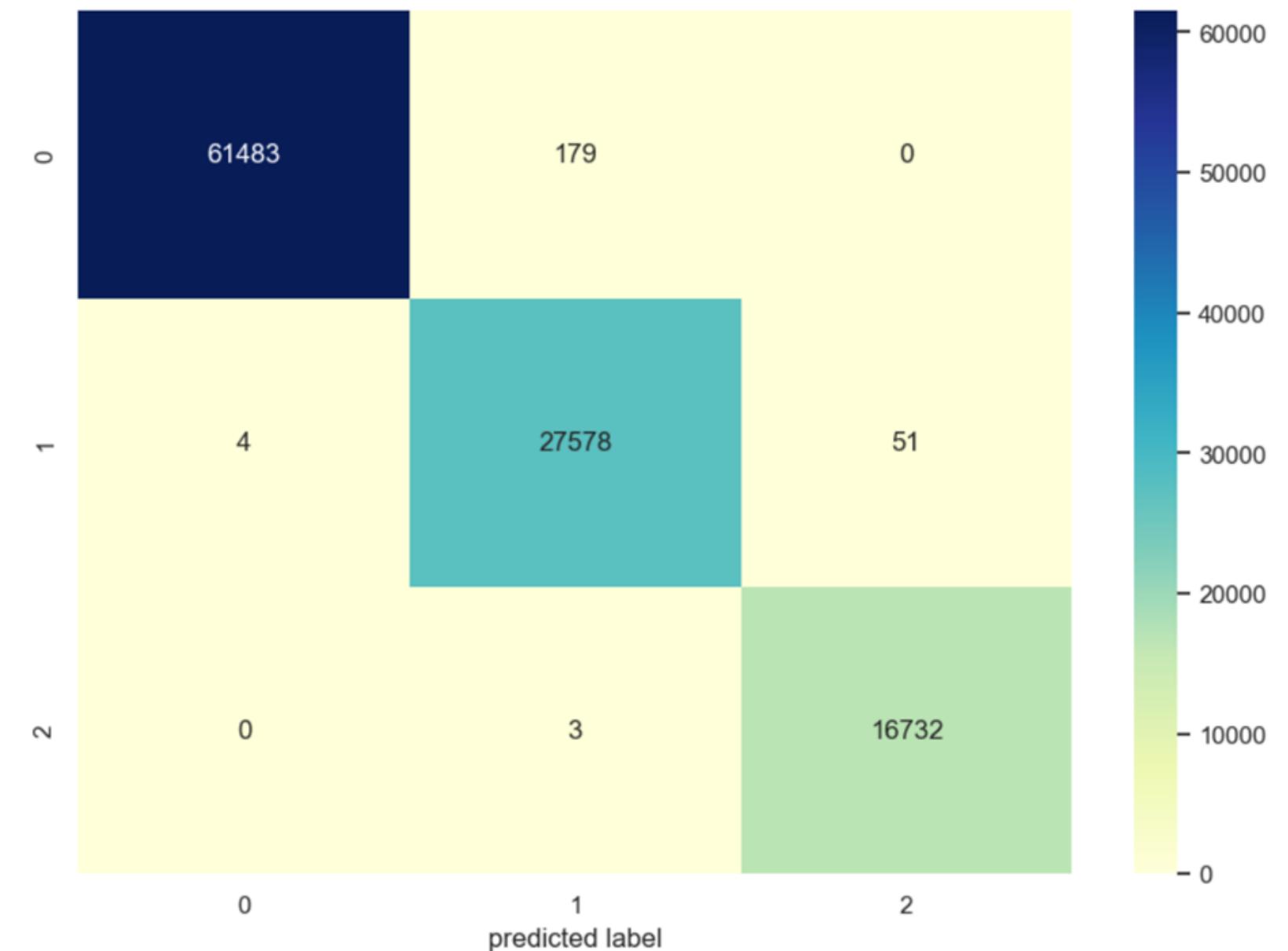


# Model 생성 및 평가

## ▶ 최적의 hyperparameter 적용 후 prediction 진행

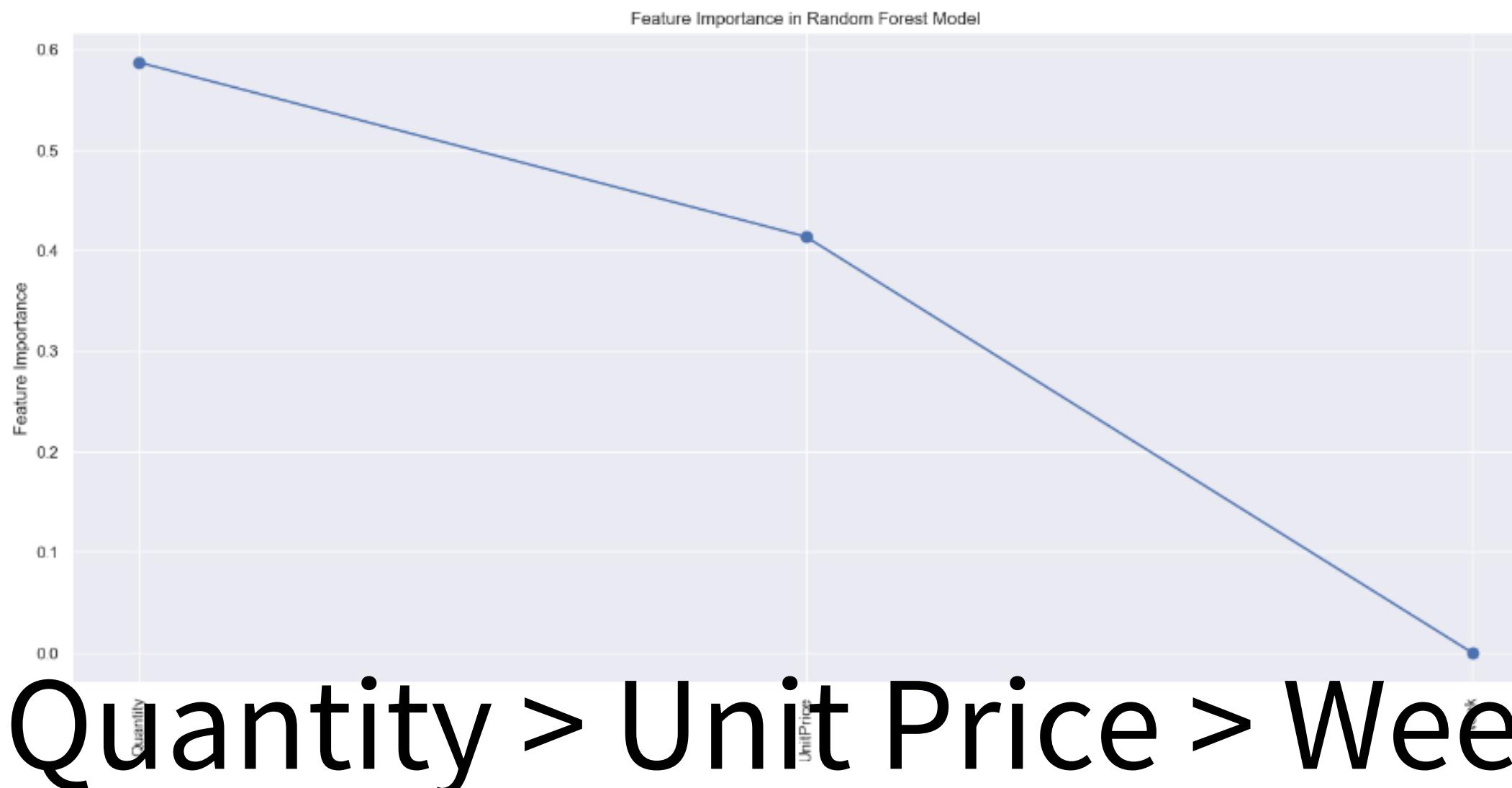
- 정확도: 0.9977 으로 매우 높음
- Precision: [0.99993495  
0.9934438 0.99696121]
- Recall: [0.99709708  
0.99800963 0.99982073]

→ 학습이 매우 잘 이루어짐



# Model 생성 및 평가

## ▶ Feature Importace 확인



## 결론

### ▶ 데이터 전처리

- elbow method & k-means clustering 통한 spend\_label 개수 설정 후 적용
- 학습하기 전 불필요한 값, 변수들 처리하여 학습 효율 향상

### ▶ Random Forest 모델 적용

- 열을 선정하여 해당 구매 정보의 금액을 3단계로 나타낸 spend\_label 예측
- Randomized Search CV 이용하여 최적의 hyperparameter 선정
- cross validation 활용해 과적합 방지 & hyperparameter tuning
- Quantity 항목이 가장 중요한 변수

# THANK YOU