@ Machine learning is the process

* Make decisions form data

@ Unsupervised learning

* Uncovering hidden patterns from unlabled data
* Ex) distinct categories (Clustering)

@ Supervised learning

* The predicted values are known
* Classification
* Regression (continuous)

\*Feature = predictor variable = independent variable

\*Target variable = dependent variable = response variable

@ Before you use supervised learning

* No missing values, numeric, pandas DataFrame or NumPy array

@ Scikit-learn syntax

**from sklearn.module import Model**

**model = Model() #instanciate model**

**model.fit(X,y) # feature array X fit to target variable array y**

**predictions = model.predict(X\_new)**

**print(predictions)**

**The classification challenge**

@classifying labels of unseen data

1. Build a model
2. Learns from the labeled data
3. Pass unlabeled data to ~ as input
4. Model predicts the labels

\*Labeled data = training data

@ k-Nearest Neighbors

KNN uses majority voting, which makes predictions based on what label the majority of nearest neighbors have. (다수의 가까운 이웃의 레이블 이용하여 예측)

@Classifiying labels of unseen data

Learns from the labeled data -> pass unlabeled data as input -> predict

@K-Nearest Neighbors

* Looking at the k closest labeled data points (가까운 이웃 기반 다수결 투표 사용)

**From sklearn.neighbors import KNeighborsClassifier**

**X = churn\_df[[”total\_day\_charge”, “total\_eve\_charge”]].values #split data into X ( 2D array)**

**Y = churn\_df[”chrun”].vaues #target’s 1D 이탈상태를 의미**

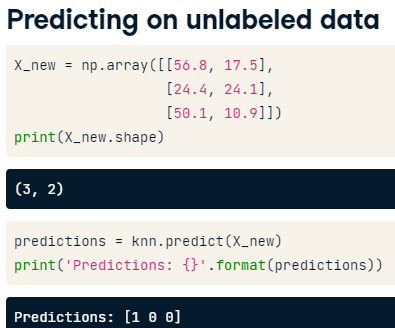
**#각 열은 기능, 각 행은 다른 관찰**

**Print(X.shape, y.shape)**

**===> (3333,2 ), (3333, )**

**Knn = KNeighborsClassifier(n\_neighbors=15)**

**Knn.fit(X, y) #특성 값X 및 대상 값 y 전달**

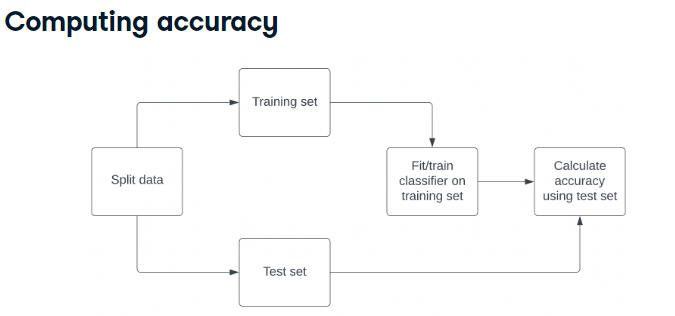


두 번째 , 세 번째 관찰 ‘이탈 없음’ 0

--- 모델이 올바른 예측을 하고 있는지 ?

Valuate the performance

Accuracy = correct predictions / total observations



**From sklearn.model\_selection import train\_test\_split**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=21, stratify=y)**

# Using the same number when repeating this step allows us to reproduce the exact split and our downstream results. It is best practice to ensure our split reflects the proportion of labels in our data.

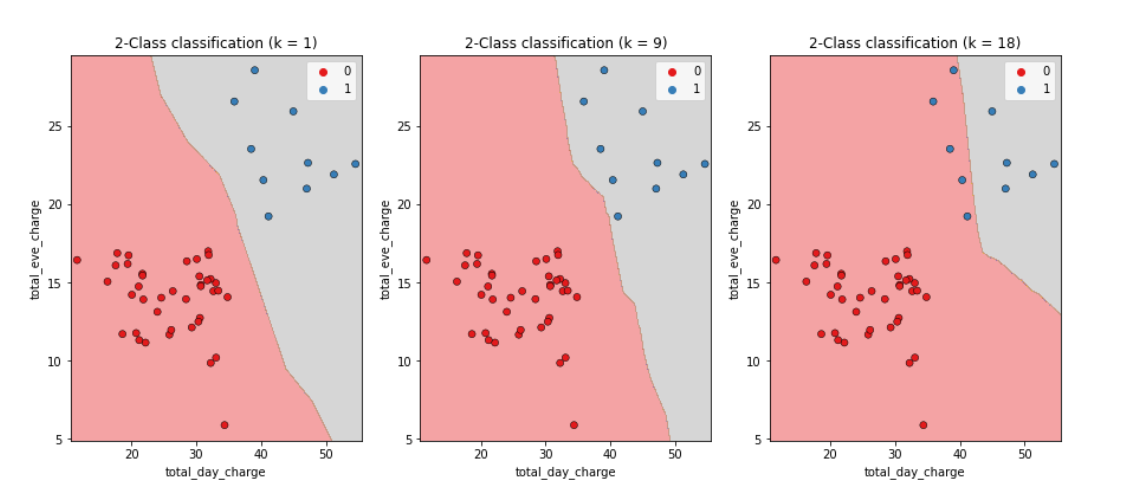
데이터 나눌 때 같은 번호 사용하면 정확한 분할과 다운 스트림 결과 사용.

훈련데이터, 테스트 데이터, 훈련 레이블 및 테스트 레이블 의 4가지 반환.

**Knn = KNeighborsClassifier(n\_neighbors = 6)**

**Knn.fit (X\_trainl , y\_train)**

**Print(knn.score(X\_test, y\_test))**



K = 모델이 관찰에 할당하는 레이블 결정하기 위한 임계값.

==> k가 커질 수록 간단, 개별 관찰의 영향을 덜 받는다 <과소적합>

\*Larger k = less complex = underfitting

\*Smaller k = more complex = overfitting