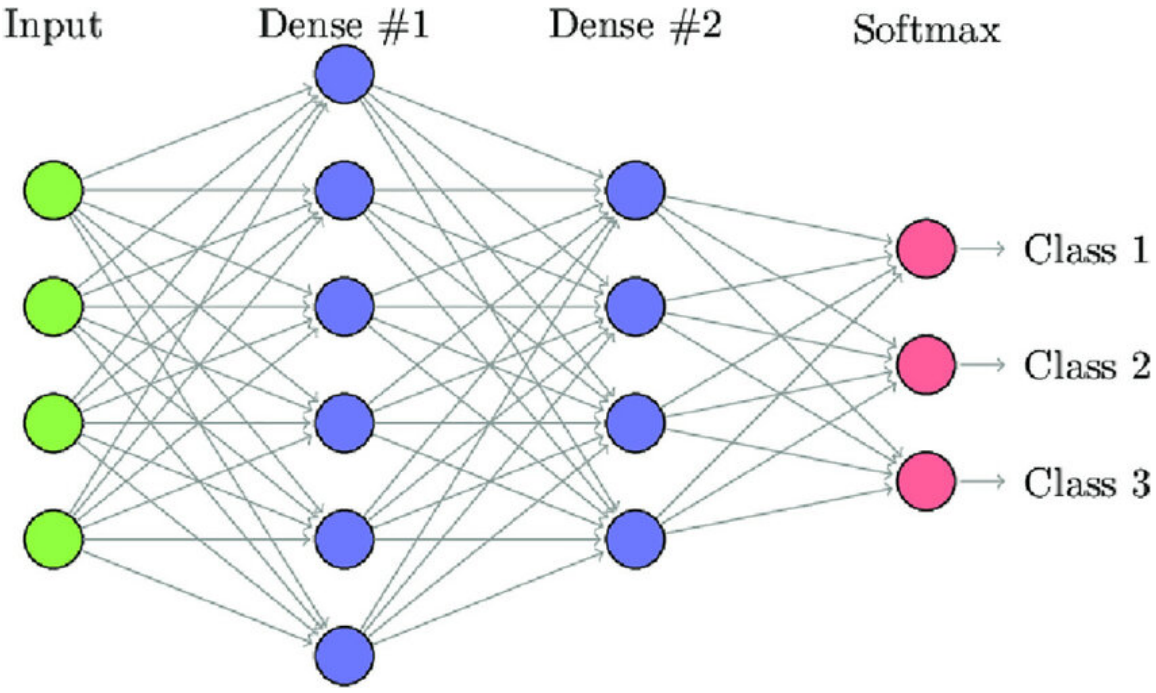
모델 제작 과정

모델 제작 과정 및 시행착오

초기 모델: Fully Connected Neural Network (FCNN)

stainless fully model.ipynb

- **선정 이유**: CSV 파일이 1차원 배열로, 각 데이터당 400개의 파장 값을 가지므로 FCNN이 적합하다고 판단했습니다.



- **구현 결과**:
  - FCNN을 사용하여 학습한 결과, 약 90%의 정확도를 달성했습니다.
  - 그러나 210 등급의 스테인리스 팬을 제대로 분류하지 못하는 문제를 발견했습니다.

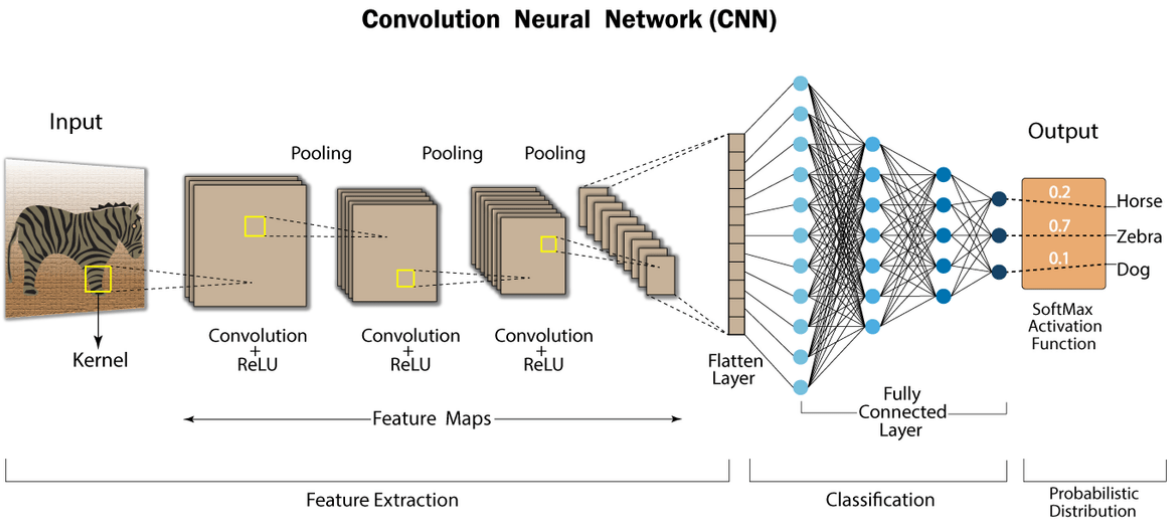
```
25/25 [==============================] - 0s 2ms/step
Accuracy: 0.90625
Confusion Matrix:
[[144   0  65   0]
 [  0 209   1   3]
 [  0   0 194   0]
 [  0   6   0 178]]
```

- 모델의 성능을 향상시키기 위해 FCNN의 층 수를 3층에서 5층으로 늘려보았지만, 결과는 크게 개선되지 않았습니다.

모델 전환: Convolutional Neural Network (CNN)

stainless CNN model.ipynb

- **전환 이유**: FCNN의 한계를 극복하고 데이터의 패턴을 더 잘 학습할 수 있는 모델이 필요했습니다.



- **CNN 도입**:
  - **CSV 데이터 재구성**:
    - CSV 파일의 1차원 배열 데이터를 CNN에 적합한 형태로 변환하기 위해, 데이터를 4차원으로 재구성했습니다.

- `X_reshaped = X_scaled.reshape(-1, 4, 100, 1)` 코드를 사용하여, 4000개의 데이터를 (4, 100) 크기의 흑백 이미지 형태로 변환했습니다.

CNN 모델 구성

- **Conv2D(32, kernel_size=(2, 2), activation='relu', input_shape=(4, 100, 1))**:
  - 2D 합성곱 레이어로, 32개의 필터를 사용해 입력 데이터의 특징을 추출합니다.
  - 커널 크기는 (2, 2)이며, 활성화 함수로 ReLU를 사용합니다.
- **MaxPooling2D(pool_size=(2, 2), padding='same')**:
  - 최대 풀링 레이어로, (2, 2) 크기의 풀링 윈도우를 사용해 입력 데이터의 크기를 줄이고 중요한 특징을 유지합니다.
  - 패딩을 'same'으로 설정하여 입력과 출력의 크기가 같도록 합니다.
- **Conv2D(64, kernel_size=(2, 2), activation='relu')**:
  - 두 번째 2D 합성곱 레이어로, 64개의 필터를 사용해 더욱 복잡한 특징을 추출합니다.
  - 커널 크기와 활성화 함수는 동일합니다.
- **Flatten()**:
  - 다차원 배열을 1차원으로 변환하여 Dense 레이어에 입력할 수 있도록 합니다.
- **Dense(128, activation='relu')**:
  - 완전 연결 레이어로, 128개의 뉴런을 사용해 특징을 학습합니다.
  - 활성화 함수로 ReLU를 사용해 비선형성을 추가합니다.
- **Dropout(0.3)**:
  - 과적합을 방지하기 위해 30%의 뉴런을 무작위로 비활성화합니다.
- **Dense(4, activation='softmax')**:
  - 출력 레이어로, 4개의 뉴런을 사용해 각 등급(201, 304, 316, 430)을 분류합니다.
  - 소프트맥스 활성화 함수를 사용해 클래스별 확률을 출력합니다.

이러한 구성은 데이터의 패턴을 더 잘 학습하고, 모든 등급의 분류 성능을 개선하는 데 도움을 주었습니다.

```
Accuracy: 0.9900000095367432
25/25 [==============================] - 0s 2ms/step
Confusion Matrix:
[[207   0   2   0]
 [  0 208   2   3]
 [  0   0 194   0]
 [  0   1   0 183]]
```