Multi Class and Multi Label

PyTorch

- Sequential
- Multi Class
- Multi Label

במקום להגדיר כל שכבה בנפרד בתוך

__init__ובמקום לקרוא להן ידנית ב-

forward()

Sequential **יוצר רצף של שכבות**

נוירונים שפועלות אחת אחרי השנייה

אוטומטית .

```python
class SimpleNN(nn.Module):
    def __init__(self):
        super().__init__()
        self.fc1 = nn.Linear(2, 16)  # שכבה ראשונה
        self.fc2 = nn.Linear(16, 8)  # שכבה שנייה
        self.fc3 = nn.Linear(8, 3)   # שכבה אחרונה
        self.relu = nn.ReLU()        # פונקציית אקטיבציה

    def forward(self, x):
        x = self.fc1(x)
        x = self.relu(x)
        x = self.fc2(x)
        x = self.relu(x)
        x = self.fc3(x)
        return x
```

```python
model = nn.Sequential(
    nn.Linear(2, 16),  # 16 → 2 :שכבה ראשונה
    nn.ReLU(),
    nn.Linear(16, 8),  # 8 → 16 :שכבה שנייה
    nn.ReLU(),
    nn.Linear(8, 3)    # 3 → 8 :שכבה אחרונה
)
```

Sequential

מצב	האם צריך Sequential או class	דוגמה
מודל פשוט עם שכבות ליניאריות רציפות	Sequential	Fully Connected (Dense)
רשת מורכבת עם חיבורים מותאמים אישית	class (nn.Module)	CNN, RNN
צריך שליטה על ()forward	class (nn.Module)	שילוב שכבות מיוחדות

מתי לבחור Sequential?

Multi-Class Classification

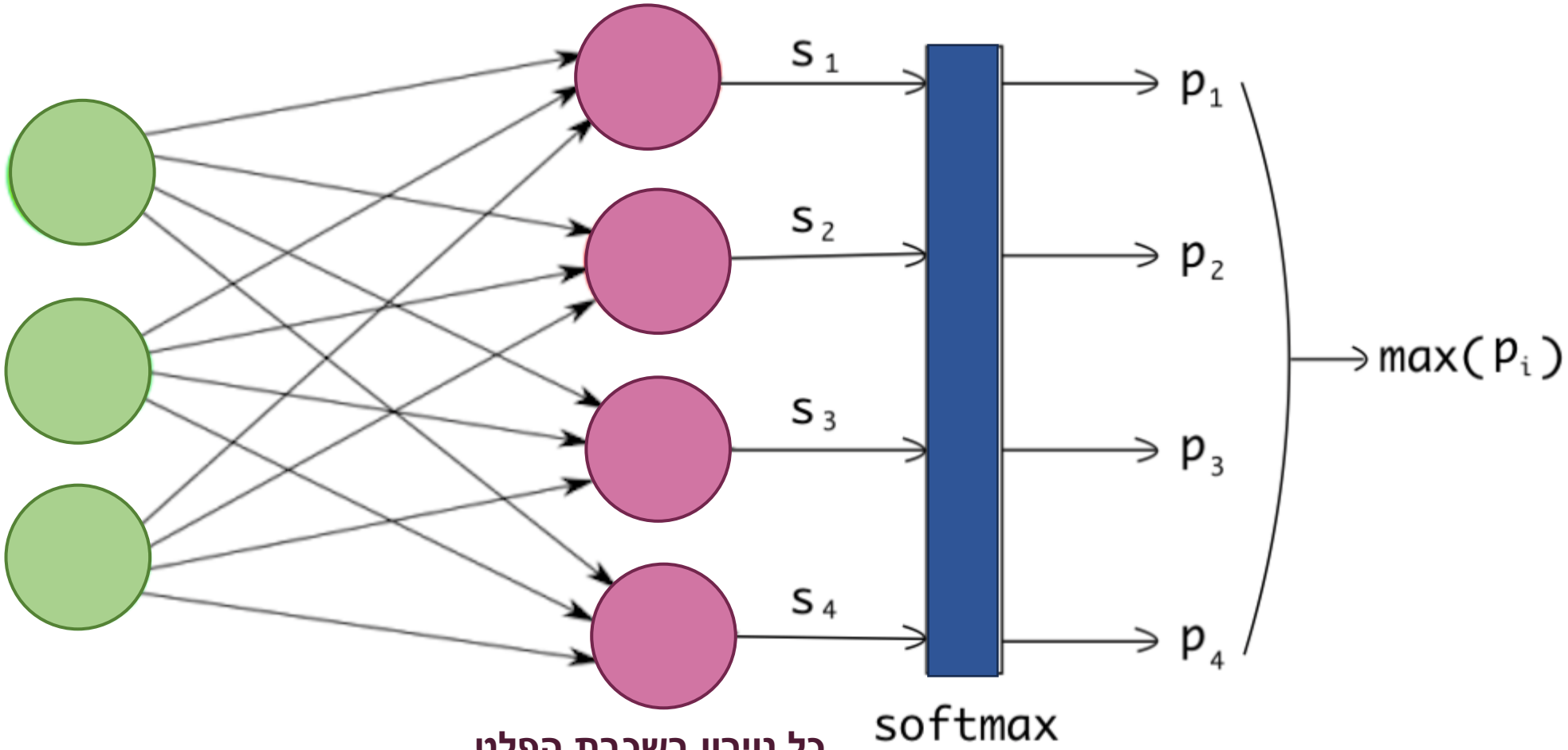
בבעיית סיווג רב-קטגורי (Multi-Class) כל דוגמה מסווגת בדיוק למחלקה **אחת** מתוך **כמה אפשריות** (3 ומעלה)

לדוגמה, אם מודל צריך לסווג תמונה לחתול, מפלצת או ילדה יפה – הוא יחזיר הסתברות לכל קטגוריה, אך התוצאה תהיה רק אחת מהן.

מספר הנוירונים בשכבת הפלט

במודל Multi-Class, **מספר הנוירונים** בשכבת הפלט **שווה למספר המחלקות**.

לדוגמה, אם יש **3 קטגוריות**, נשתמש ב **3-נוירונים ביציאה**, וכל אחד מהם ייצג **את ההסתברות** למחלקה מסוימת.



כל נוירון בשכבת הפלט
מייצג מחלקה אחת

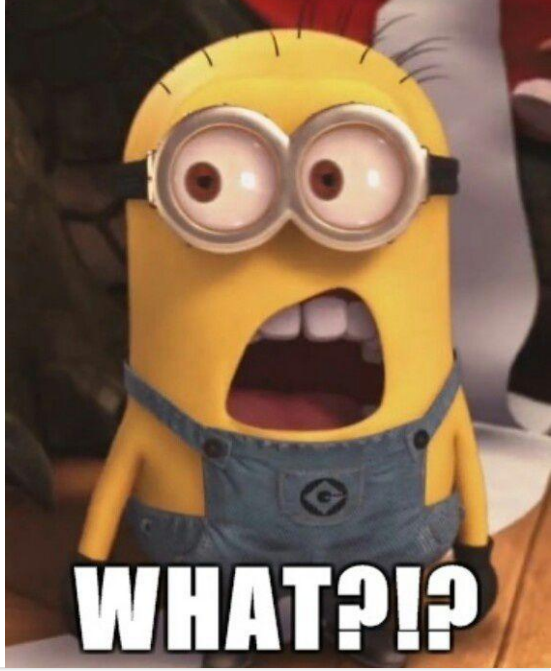
שכבת
אקטיבציה
בוחרת את
המחלקה עם
הסיכוי הכי גבוה

argmax היא פונקציה שמחזירה את האינדקס של הערך הגדול ביותר במערך נתון.
כלומר, היא בוחרת את התגית עם הערך הגבוה ביותר ומחזירה את מיקומה, בעוד שהשאר מתבטלים.
הבעיה היא ש- argmax אינה ניתנת לגזירה, מכיוון שהיא לא מחזירה ערכים רציפים אלא בחירה חד-משמעית (0 או 1 לכל תגית).
לכן, הנגזרת שלה היא תמיד אפס, מה שאומר שלא ניתן לחשב גרדיאנט ולבצע עדכון משקלים במהלך האימון של רשת נוירונים.

כדי לפתור את זה, **Softmax** ממירה את הפלט של הרשת להתפלגות הסתברותית, כך ש:
כל הערכים יהיו בין 0 ל-1.
סכום כל ההסתברויות יהיה שווה ל-1.
הערך הגבוה ביותר עדיין מצביע על התגית הנבחרת, אבל עכשיו גם יש שיפועים שניתן לגזור ולשפר באמצעות גרדיאנט.
בזכות זה, Softmax מאפשר לרשת ללמוד ולשפר את הביצועים שלה בצורה הדרגתית לאורך האימון.

Softmax

$$s\left(x_i\right) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$



מה זה עושה תבלס

```
# איך עובד softmax
x = torch.tensor([1.0,1.5,3.])  # טנסור של שלושה מספרים
```

```
x.argmax()#  מחזיר את המיקום של הערך המקסימלי
```

```
tensor(2)
```

```
x.softmax(dim =0)
```

```
tensor([0.0996, 0.1643, 0.7361])
```

CrossEntropyLoss

זוהי פונקציית הפסד שמודדת כמה ההסתברות שחזינו קרובה להתפלגות האמיתית.

- אם המודל בטוח ונתן הסתברות גבוהה למחלקה הנכונה, האובדן נמוך.
- אם המודל טועה ונתן הסתברות גבוהה למחלקה שגויה, האובדן גדול מאוד.
- כבר כולל Softmax ולכן אין צורך להפעיל Softmax במודל במפורש במודל עצמו.

True probability distribution (one-shot)

$$H(p, q) = -\sum_{x \in classes} p(x) \log q(x)$$

Your model's predicted probability distribution

Multi-Label Classification

בבעיית סיווג מרובה-תוויות כל דוגמה יכולה להשתייך לכמה קטגוריות בו-זמנית, בניגוד ל- Multi-Class, שבו כל דוגמה משויכת לתגית אחת בלבד.

דוגמאות:

ז'אנר של סרט – סרט יכול להיות גם קומדיה, גם פעולה, וגם דרמה.

זיהוי אובייקטים בתמונה – תמונה יכולה להכיל גם בננה וגם מיניון וגם חללית

ניתוח טקסט – פוסט יכול להיות גם פוגעני וגם שקרי וגם מצחיק (למרות שלא יפה לצחוק במצב כזה)

Multi-Label Classification

- **מספר הנוירונים** בשכבת הפלט **שווה למספר התוויות האפשריות**, וכל נוירון מחזיר הסתברות נפרדת לכל קטגוריה.
- כל יציאה של המודל עוברת דרך Sigmoid, כך שכל תווית מקבלת הסתברות נפרדת בין 0 ל-1.
- משתמשים ב- **BCELoss (Binary Cross-Entropy)**, כי **כל תווית היא סיווג בינארי נפרד** (האם התווית קיימת או לא)

מדידת ביצועי המודל – למה Accuracy לא מספיק?

- בבעיות Multi-Label, מדד דיוק (Accuracy) לא שימושי, כי ייתכן שהמודל יזהה רק חלק מהתוויות הנכונות, אך עדיין ייחשב לטועה.
- במקום זאת, משתמשים במדדים כמו F1-score, שמודד איזון בין דיוק (Precision) לשליפה (Recall)

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$Precision = \frac{TP}{TP + FP}$$

מתוך כל מי שאמרנו עליו
"חולה" כמה מהם האמת היו
חולים
או - כמה "נפלנו" בהגיד
"חולה" למי שבריא בעצם

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

מתוך כל מי שבאמת חולה,
כמה הצלחנו לזהות.
או כמה "נפלנו" בהגיד
"בריא" למי שבעצם חולה

```
print(classification_report(y_test_nonlinear, y_pred_nonlinear))
```

	precision	recall	f1-score	support
0	1.00	0.41	0.58	22
1	0.58	1.00	0.73	18
accuracy			0.68	40
macro avg	0.79	0.70	0.66	40
weighted avg	0.81	0.68	0.65	40

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

ממוצע הרמוני

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

תזכורת
classification report

פונקציית הפסד	מטריקת פלט	סוג בעיה
CrossEntropyLoss	Softmax → Argmax	Multi-Class
BCELoss או BCEWithLogitsLoss	Sigmoid (לכל תווית בנפרד)	Multi-Label