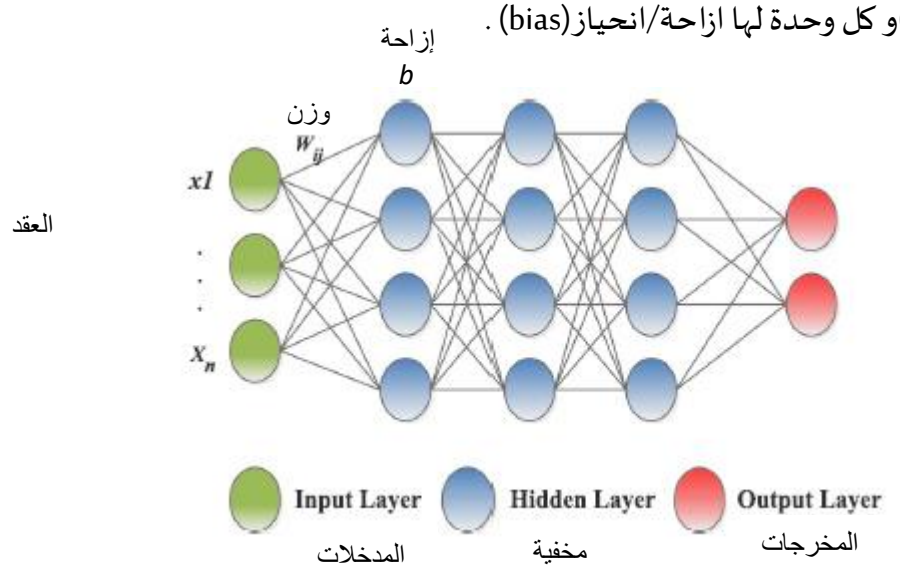
المحور الثاني : الشبكات العصبية الأمامية(Feed-Forward Neural Networks)

I. مقدمة سريعة (لماذا نحتاج MLP في إدارة الأعمال)

تُعد الشبكات العصبية متعددة الطبقات (MLP) من الأدوات الذكية المتقدمة في تحليل البيانات المعقدة، إذ تمتاز بقدرتها على تعلّم العلاقات غير الخطية بين المتغيرات. في مجال إدارة الأعمال، تساعد MLP على تحليل سلوك الأسواق والعملاء، وتوقّع المبيعات، وتحسين قرارات التسعير والتسويق، ما يجعلها أداة فعالة لدعم القرارات الإدارية الاستراتيجية. ومن أبرز تطبيقاتها: التنبؤ بالطلب المستقبلي على المنتجات (انحدار)، وتصنيف العملاء حسب درجة الولاء أو المخاطر (تصنيف ثنائي أو متعدد)، وتحليل العوامل المؤثرة في الأداء المالي للمؤسسة. يساهم هذا النهج في تعزيز كفاءة التخطيط واتخاذ القرار، وتحقيق ميزة تنافسية قائمة على الذكاء الاصطناعي.

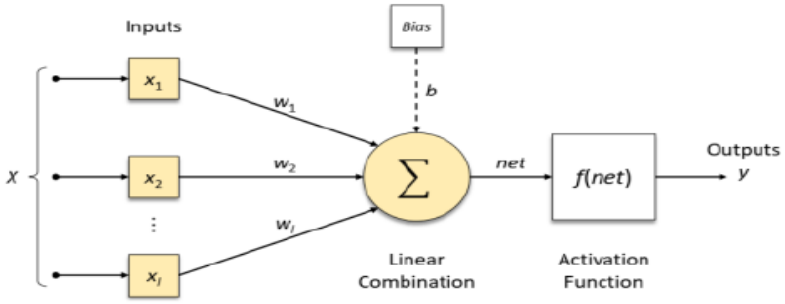
II. مفاهيم أساسية

- **شبكة عصبية أمامية (Feed-Forward)** : لا توجد دورات (loops). البيانات تدخل من طبقة المدخلات وتمر خلال طبقات مخفية إلى طبقة المخرج.
- **MLP (Multi-Layer Perceptron)** : يتكون من طبقة إدخال، عدد واحد أو أكثر من الطبقات المخفية (كلها تحتوي على وحدات عصبية) ، وطبقة إخراج. كل وصلة بين وحدتين لها وزن (weight)و كل وحدة لها ازاحة/انحياز(bias) .



III. العنصر الأساسي: الخلية العصبية(Neuron / Perceptron)

اخترع بيرسيبترون بواسطة فرانك روزنبلات عام 1957 مختبر كورنيل للطيران، بيرسبترون هو أبسط شكل للشبكة العصبية الاصطناعية و هو عبارة عن مصنف ثنائي. إن بُنية هذه الشبكة العصبية ليست سوى طبقة إدخال واحدة بمخرج واحد فقط، ومن ثم يطلق عليها أيضًا اسم الشبكة العصبية أحادية الطبقة.



كما يمكن رؤيته، هناك عدد كبير من المدخلات في هذه الشبكة، والتي يتنبأ مجموعها، بعد الحساب، بالإخراج باستخدام دالة التنشيط. مع قائمة المدخلات . x={$x1,x2,...,xn$} سيكون لكل إدخال متجه وزن . w={$w1,w2,...,wn$} يتم حساب مجموع الاوزان بالمعادلة :

$$net = \sum_{i=0}^{n} wi * xi + b$$

ثم تحصل دالة التنشيط على الناتج y بناءً على العتبة θ :

$$y = f(net) = \begin{cases} 1 \; if \; net \geq \theta \\ 0 \; if \; net < \; \theta \end{cases}$$

أمثلة دوال تنشيط:

- **Sigmoid** : مفيدة لتصنيف ثنائي (لكن لها مشاكل التشبع).

$$\sigma(x) = sigmoid(x) = \frac{1}{1 + e^{-x}}$$

- **ReLU** : شائعة في الطبقات المخفية.

$$ReLU(x) = \max(0, x)$$

- **Softmax** (للمخرجات متعددة الفئات): تحول المتجه إلى احتمالات.

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

IV. خوارزمية تعلم بيرسيبترون(Perceptron Learning Algorithm)

1. المعادلة الأساسية

تُحسب الإشارة الداخلة إلى الخلية العصبية كما يلي:

z = w1 * x1 + w2 * x2 + ... + wn * xn + b

ثم نحصل على الخرج (التنبؤ) بواسطة دالة التنشيط:(Step Function)

$$\hat{y} = \begin{cases} 1 \; if \; z \geq 0 \\ 0 \; if \; z < 0 \end{cases}$$

2. قاعدة التعلم (تحديث الأوزان)

بعد كل عينة تدريبية، نقوم بتحديث الأوزان والانحياز استنادًا إلى الخطأ بين القيمة الحقيقية والتنبؤ:

خطأ العينة:

$$e = y - \hat{y}$$

تحديث الأوزان:

$$wj = wj + \eta * e * xj$$

تحديث الانحياز:

$$b = b + \eta * e$$

حيث η تمثل **معدل التعلم**.**(Learning rate)**

3. خوارزمية التعلم خطوة بخطوة

1. نهيئ الأوزان والانحياز بقيم صغيرة.
2. لكل عينة تدريبية:(x,y)

- نحسب $\hat{y} = f(w \cdot x + b)$
- نحسب الخطأ $e = y - \hat{y}$
- نحدّث الأوزان والانحياز كما في المعادلات أعلاه.

3. نكرر حتى يتناقص الخطأ أو نصل إلى عدد محدد من التكرارات(epochs) .

❖ **مثال رقمي: تعلم دالة AND باستخدام البيرسيبترون**

1) جدول الحقيقة(Truth Table)

x1	x2	y (الخرج الحقيقي)
0	0	0
0	1	0
1	0	0
1	1	1

الهدف هو أن يتعلّم البيرسيبترون إنتاج القيمة **1فقط عندما يكون x1 = 1 و x2 = 1 .**

2) التهيئة الأولية

نبدأ بـ:

$w1 = 0$

$w2 = 0$

$b = 0$

$\eta = 0.1$ (معدل التعلم)

3) الخطوات الحسابية

العينة الأولى(x1=0, x2=0, y=0) :

$$z = (w1 * x1) + (w2 * x2) + b$$
$$z = (0 * 0) + (0 * 0) + 0 = 0$$

$$\hat{y} = 1 \text{ if } z >= 0, \text{ otherwise } 0$$
$$\longrightarrow \hat{y} = 1$$

الخطأ:

$$e = y - \hat{y} = 0 - 1 = -1$$

تحديث القيم:

$$w1 = w1 + \eta * e * x1 = 0 + 0.1 * (-1) * 0 = 0$$
$$w2 = w2 + \eta * e * x2 = 0 + 0.1 * (-1) * 0 = 0$$
$$b = b + \eta * e = 0 + 0.1 * (-1) = -0.1$$

العينة الثانية(x1=0, x2=1, y=0) :

$$z = (w1 * x1) + (w2 * x2) + b$$
$$z = (0 * 0) + (0 * 1) + (-0.1) = -0.1$$

$$\hat{y} = 1 \text{ if } z >= 0, \text{ otherwise } 0$$
$$\longrightarrow \hat{y} = 0$$

$$e = y - \hat{y} = 0 - 0 = 0$$

⟵لا تحديث للأوزان.

العينة الثالثة(x1=1, x2=0, y=0) :

$$z = (w1 * x1) + (w2 * x2) + b$$
$$z = (0 * 1) + (0 * 0) + (-0.1) = -0.1$$

$\hat{y} = 0$ لأن$(z < 0)$

$e = 0 - 0 = 0$

⟵لا تحديث.

<u>العينة الرابعة(x1=1, x2=1, y=1) :</u>

$z = (w1 * x1) + (w2 * x2) + b$

$z = (0 * 1) + (0 * 1) + (-0.1) = -0.1$

$\hat{y} = 0$لأن$(z < 0)$

$e = y - \hat{y} = 1 - 0 = 1$

تحديث القيم:

$w1 = w1 + \eta * e * x1 = 0 + 0.1 * 1 * 1 = 0.1$

$w2 = w2 + \eta * e * x2 = 0 + 0.1 * 1 * 1 = 0.1$

$b = b + \eta * e = -0.1 + 0.1 * 1 = 0$

بعد التحديثات

القيم النهائية بعد مرور عينة التدريب الأولى:

$w1 = 0.1$

$w2 = 0.1$

$b = 0$

◆ اختبارالنموذج بعد التعلم

x1	x2	$z = (w_1x_1 + w_2x_2 + b)$	$\hat{y}$	النتيجة
0	0	0	1	✗ خطأ
0	1	0.1	1	✗ خطأ
1	0	0.1	1	✗ خطأ
1	1	0.2	1	✓ صحيح

بعد تكرار التدريب(عدة epochs) ، ستستقر القيم على حدود تفصل بين الفئتين بدقة أعلى.

V. الانتقال إلى الشبكات العصبية متعددة الطبقات(MLP)

لماذا نحتاج أكثر من طبقة؟

البيرسيبترون البسيط يستطيع فقط حل **المشكلات الخطية** مثل AND و OR ، لكنه يفشل في المشكلات **غير الخطية** مثل XOR.
لحل ذلك، نضيف **طبقات مخفية (Hidden Layers)** بين طبقة الإدخال وطبقة الإخراج، بحيث تتعلم هذه الطبقات تمثيلات داخلية أكثر تعقيدًا.

1. بنية الشبكة العصبية متعددة الطبقات(MLP)

تتكون من:

1. **طبقة الإدخال:(Input Layer)**
   تمثل المتغيرات (features) الداخلة للنموذج.
2. **طبقات مخفية:(Hidden Layers)**
   تحتوي على وحدات عصبية تستخدم دوال تنشيط مثل ReLU أو Sigmoid للتعلم غير الخطي.
3. **طبقة الإخراج:(Output Layer)**
   تنتج التنبؤ النهائي — تصنيف (1/0 أو متعدد الفئات) أو قيمة رقمية (في الانحدار).

2. خوارزمية التعلم في MLP :الانتشار العكسي(Backpropagation)

الـ **Backpropagation**هي الطريقة المستخدمة لتحديث الأوزان في الشبكات متعددة الطبقات.
الفكرة الأساسية:

1. نحسب التنبؤ (Forward Pass).
2. نحسب الخطأ بين التنبؤ والقيمة الحقيقية.
3. نحسب مشتقات الخطأ بالنسبة للأوزان(Gradient).
4. نحدث الأوزان باستخدام خوارزمية **الانحدار التدرّجي(Gradient Descent).**

❖ **مثال عملي باستخدامKeras**

في هذا المثال، سنبني شبكة **MLPبسيطة** لتصنيف بيانات منطق **AND**نفسها، ولكن باستخدام بنية متعددة الطبقات.

كود بايثون باستخدامKeras

```
#استيراد المكتبات
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import SGD

# ANDبيانات تدريب دالة
X = np.array([[0,0], [0,1], [1,0], [1,1]])  #المدخلات
y = np.array([[0], [0], [0], [1]])        #المخرجات الحقيقية

# MLPبناء نموذج
model = Sequential([
    Dense(4, input_dim=2, activation='relu'),  #طبقة المدخلات
    Dense(1, activation='sigmoid')          #طبقة إخراج للتصنيف الثنائي
])

#تجميع النموذج
model.compile(
  optimizer=SGD(learning_rate=0.1),
  loss='binary_crossentropy',
  metrics=['accuracy']
)

#تدريب النموذج
model.fit(X, y, epochs=500, verbose=0)

#اختبار النموذج
predictions = model.predict(X)
print("Predictions:")
```

```
for i, p in enumerate(predictions):
  print(f"Input: {X[i]} => Predicted: {p[0]:.4f} (Rounded: {int(round(p[0]))})")
```

تفسير النتائج

بعد عدة تكرارات(epochs) ، سيتعلم النموذج أن:

x1	x2	الناتج المتوقع
0	0	0
0	1	0
1	0	0
1	1	1

تمامًا مثل **دالةAND** ، ولكن هذه المرة تم التعلم باستخدام شبكة متعددة الطبقات ودوال تنشيط غير خطية، مما يجعلها قادرة على حل مشكلات أعقد لاحقًا.

VI. الخاتمة

- البيرسيبترون هو أساس الشبكات العصبية لكنه محدود بالمشكلات الخطية.
- MLPتتفوق بفضل وجود طبقات مخفية ودوال تنشيط غير خطية.
- خوارزمية **Backpropagation**تسمح للنموذج بتعلّم العلاقات المعقدة بين المدخلات والمخرجات.
- باستخدام مكتبة **Keras**، يمكننا بسهولة بناء وتدريب نماذج MLP لتطبيقات في **التنبؤ**، **التصنيف**، **التسويق**، **وتحليل البيانات** في مجال إدارة الأعمال.