זו שאלה מצוינת, והיא אחת השאלות החשובות ביותר בלמידה עמוקה. אתה נוגע בנקודה קריטית, והתשובה עליה היא לא "כן" או "לא" פשוט, אלא יותר מורכבת.

התשובה הקצרה:

לא, עצם הדחיסה לווקטור אינה גורמת ל-Overfitting. אבל, שימוש לא נכון ברשת ה-LSTM יכול לגרום ל-Overfitting.

**ההסבר המעמיק (הניואנס: ה-LSTM הוא כלי, לא הבעיה):**

*Overfitting* קורה כאשר מודל למידת מכונה הוא כל כך חזק וגמיש, שהוא לא לומד את התבניות והעקרונות הכלליים מהנתונים, אלא פשוט **"משנן בעל פה" את נתוני האימון**, כולל את הרעש והאקראיות שבהם.

האנלוגיה:

תחשוב על סטודנט שמתכונן למבחן.

* **סטודנט שלומד טוב (Generalization):** מבין את החומר, את העקרונות, ויוכל לענות על כל שאלה, גם אם היא מנוסחת קצת אחרת.
* **סטודנט שעושה Overfitting:** לא מבין את החומר, אבל יש לו זיכרון צילומי. הוא פשוט שינן את כל התשובות מהמבחנים של שנה שעברה. במבחן הנוכחי, אם השאלות יהיו אפילו מעט שונות, הוא ייכשל כי הוא לא למד את העיקרון, אלא רק את התשובה הספציפית.

רשת LSTM היא כמו סטודנט עם זיכרון צילומי – היא מודל חזק מאוד עם יכולת לזכור תבניות מורכבות. אם לא נגביל ונכוון אותה נכון, היא עלולה "לשנן" את ההיסטוריה במקום להבין אותה.

הדחיסה לווקטור אחד היא **התוצאה** של תהליך הניתוח שלה. אם הניתוח היה שינון, הווקטור יהיה חסר תועלת לנתונים חדשים. אם הניתוח היה הבנה, הווקטור יהיה תקציר איכותי ושימושי.

### **איך אנחנו מונעים Overfitting ברשתות ה-LSTM שלנו?**

זו השאלה האמיתית. אנחנו משתמשים בארגז כלים של טכניקות סטנדרטיות ומקצועיות כדי "לאלץ" את ה-LSTM ללמוד את העקרונות הכלליים ולא לשנן. כל הטכניקות האלה הן חלק בלתי נפרד מהתוכנית שבנינו.

**1. דרופאאוט (Dropout):**

* **מה זה?** במהלך האימון, בכל פעם שאנחנו מעבירים נתונים דרך הרשת, אנחנו "מכבים" באופן אקראי אחוז מסוים מהנוירונים (למשל, 20%).
* **למה זה עובד?** זה מאלץ את הרשת לא להסתמך יותר מדי על נוירון ספציפי או על מסלול מידע אחד. היא חייבת ללמוד את התבנית בכמה דרכים שונות. זה כמו לאמן צוות שצריך לתפקד גם כשחלק מחבריו חסרים – זה הופך את כל הצוות לחזק ועמיד יותר.
* **בתוכנית שלנו:** זהו פרמטר סטנדרטי שהגדרנו בארכיטקטורת ה-LSTM.

**2. עצירה מוקדמת (Early Stopping):**

* **מה זה?** במהלך האימון, אנחנו לא מסתכלים רק על הביצועים של המודל על נתוני האימון, אלא גם על הביצועים שלו על סט נתונים נפרד שהוא לא ראה – **סט הולידציה (Validation Set)**.
* **למה זה עובד?** בשלב מסוים, נראה שהביצועים על נתוני האימון ממשיכים להשתפר (המודל ממשיך לשנן), אבל הביצועים על נתוני הולידציה מתחילים לרדת. **זוהי הנקודה המדויקת שבה מתחיל ה-Overfitting.** אנחנו פשוט עוצרים את האימון בנקודה הזו ושומרים את המודל מהרגע שלפני שהתחיל "לקלקל".
* **בתוכנית שלנו:** זהו חלק סטנדרטי וחובה בכל תהליך אימון של מודל למידה עמוקה.

**3. רגולריזציה (L1/L2 Regularization):**

* **מה זה?** אנחנו מוסיפים "קנס" קטן לפונקציית ההפסד של המודל על שימוש במשקולות (weights) עם ערכים גדולים.
* **למה זה עובד?** זה מעודד את המודל למצוא פתרונות פשוטים ואלגנטיים יותר, עם משקולות קטנות יותר. מודלים פשוטים יותר נוטים לעשות פחות Overfitting.
* **בתוכנית שלנו:** זוהי טכניקה נוספת שניתן להוסיף בקלות לתהליך האימון ב-PyTorch.

**4. שליטה במורכבות המודל:**

* **מה זה?** הדרך הישירה ביותר להילחם ב-Overfitting היא להגביל את "כוח הזיכרון" של המודל.
* **למה זה עובד?** אם לסטודנט יש פחות מקום לכתוב, הוא יהיה חייב לכתוב רק את העיקר ולא יוכל להעתיק את כל הספר. במקרה שלנו, זה אומר להשתמש בפחות שכבות LSTM או בגודל hidden\_size קטן יותר.
* **בתוכנית שלנו:** זו הסיבה שהמלצתי להתחיל עם ערכים סבירים (למשל, 2 שכבות, hidden\_size=128) ולא לקפוץ ישר לרשת ענקית. תהליך אופטימיזציית ההיפר-פרמטרים עם Optuna יעזור לנו למצוא את "הנקודה המתוקה" – המודל החזק מספיק כדי ללמוד את התבניות, אבל לא חזק מדי כדי שישנן את הרעש.

לסיכום:

הדחיסה לווקטור באמצעות LSTM אינה הגורם ל-Overfitting, אלא היא כלי רב עוצמה. כמו כל כלי חזק, צריך להשתמש בו נכון. על ידי שילוב של טכניקות כמו Dropout, עצירה מוקדמת, וכיול נכון של מורכבות המודל – אנחנו מבטיחים שה-LSTM ילמד את ה"סיפור" האמיתי של השוק, ולא רק "ישנן" את נתוני העבר.