**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**
**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**
**KHOA TOÁN – TIN HỌC**

# BÁO CÁO ĐỒ ÁN

# PYTHON CHO KHOA HỌC DỮ LIỆU

## ĐỀ TÀI

# DỰ ĐOÁN NGUY CƠ MẮC CÁC VẤN ĐỀ LIÊN QUAN ĐẾN SỨC KHỎE TINH THẦN

**Giảng viên hướng dẫn:** Hà Minh Tuấn

**Sinh viên thực hiện:**

Nguyễn Thị Kim Hằng 23280056

Nguyễn Thị Xuân Dung 23280048

Nguyễn Thị Khánh Ly 23280069

*TP. Hồ Chí Minh, tháng 12 năm 2025*

**MỤC LỤC**

# A. GIỚI THIỆU BÀI TOÁN VÀ DỮ LIỆU

## I. Giới thiệu bài toán và dự án

Sức khỏe tinh thần đang trở thành một trong những thách thức y tế và xã hội nghiêm trọng trên toàn cầu. Theo Tổ chức Y tế Thế giới (WHO), hiện có gần một tỷ người trên thế giới đang phải chung sống với các rối loạn tâm thần, ảnh hưởng trực tiếp đến chất lượng cuộc sống, năng suất lao động và sự phát triển bền vững của xã hội. Các vấn đề phổ biến như trầm cảm, lo âu ngày càng gia tăng, đặc biệt trong bối cảnh áp lực công việc, nhịp sống hiện đại và những biến động xã hội kéo dài.

Tại Việt Nam, tình hình sức khỏe tinh thần cũng đang ở mức đáng báo động. Theo các thống kê gần đây, khoảng 15% dân số mắc các rối loạn tâm thần phổ biến, trong đó có khoảng ba triệu người đang phải đối mặt với rối loạn trầm cảm. Tuy nhiên, phần lớn các trường hợp chưa được phát hiện hoặc can thiệp kịp thời do hạn chế trong công tác sàng lọc, thiếu nguồn lực chuyên môn và tâm lý e ngại khi tiếp cận các dịch vụ hỗ trợ tâm lý. Điều này dẫn đến nhiều hệ quả nghiêm trọng.

Trước thực trạng đó, việc ứng dụng các phương pháp phân tích dữ liệu và mô hình học máy trong lĩnh vực sức khỏe tinh thần đang được xem là một hướng tiếp cận giàu tiềm năng. Các mô hình dự đoán có khả năng khai thác thông tin từ nhiều yếu tố khác nhau như độ tuổi, môi trường làm việc, mức độ trầm cảm, thói quen sinh hoạt và các đặc điểm xã hội, từ đó dự đoán nguy cơ mắc các vấn đề liên quan đến sức khỏe tinh thần của mỗi cá nhân.

Trong dự án này, nhóm tập trung xây dựng và đánh giá các mô hình dự đoán nguy cơ mắc các vấn đề sức khỏe tinh thần nhằm hỗ trợ quá trình sàng lọc sớm và nhận diện nhóm có nguy cơ cao. Việc dự đoán không chỉ góp phần vào việc nâng cao hiệu quả phát hiện và can thiệp sớm mà còn giúp giảm gánh nặng y tế, đồng thời cung cấp cơ sở khoa học cho việc hoạch định các chính sách chăm sóc sức khỏe tinh thần trong tương lai.

## II. Giới thiệu về bộ dữ liệu

Bộ dữ liệu này cung cấp mô phỏng tổng hợp, chân thực về phản hồi khảo sát sức khỏe tâm thần toàn cầu từ 10.000 cá nhân. Bộ dữ liệu được tạo ra để phản ánh các mô hình thực tế được thấy trong dữ liệu sức khỏe tâm thần tại nơi làm việc, đồng thời đảm bảo tính ẩn danh và quyền riêng tư hoàn toàn.

- 10.000 mẫu đại diện cho các cá nhân ẩn danh.

- Phạm vi phủ sóng toàn cầu đa dạng.

***Ghi chú:***

- Bộ dữ liệu được tạo ra bằng cách mô phỏng lại các mô hình, phân bố và xu hướng được quan sát trong các tập dữ liệu và khảo sát sức khỏe tinh thần cộng đồng.

- Phương pháp tổng hợp này cho phép dữ liệu duy trì tính chân thực và cấu trúc tương đồng với dữ liệu thực tế, đồng thời đảm bảo loại bỏ hoàn toàn mọi thông tin nhạy cảm hoặc định danh cá nhân.

***Chú thích dữ liệu:***

| Cột | Mô tả |
|---|---|
| **age** | Độ tuổi |
| **gender** | Giới tính (Male - nam, Female - nữ) |
| **employment_status** | Tình trạng việc làm |
| **work_environment** | Môi trường làm việc |
| **mental_health_history** | Có lịch sử mắc vấn đề về sức khỏe tinh thần |
| **seeks_treatment** | Có tìm đến điều trị hay không |
| **stress_level** | Mức độ căng thẳng (từ 1 đến 10) |
| **sleep_hour** | Số giờ ngủ mỗi ngày |
| **physical_activity_days** | Số ngày vận động mỗi tuần |
| **depression_score** | Điểm đo lường triệu chứng trầm cảm |
| **anxiety_score** | Điểm đo lường độ mức lo âu |
| **social_support_score** | Điểm đánh giá mức độ hỗ trợ xã hội (gia đình, bạn bè,...) |
| **productivity_score** | Đánh giá năng suất làm việc |
| **mental_health_risk** | Nguy cơ sức khỏe tinh thần |

## B. GIỚI THIỆU VỀ LỚP VÀ CÁC PHƯƠNG THỨC

### I. Tiền xử lý dữ liệu

| Tên lớp | Chức năng của lớp | Tên phương thức | Chức năng của phương thức |
|---|---|---|---|
| DataPreprocessor | Tiền xử lý dữ liệu | is_numeric | Kiểm tra một cột có phải kiểu số hay không |
| | | load | Nạp dữ liệu từ file |
| | | convert | Chuyển đổi kiểu dữ liệu phù hợp |
| | | feature_separation | Phân loại các cột (numeric và categorical) |
| | | summary | Thông tin về dữ liệu |
| | | handle_missing_ numeric | Xử lý giá trị khuyết ở cột số |
| | | handle_missing_ categorical | Xử lý missing values ở cột phân loại |
| | | normalize_ categorical | Chuẩn hóa dữ liệu cột categorical |
| | | skewness_for_ numeric_cols | Tính độ lệch của các cột số |
| | | handle_outlier | Xử lý ngoại lai |
| | | encode_categorical | Mã hóa biến phân loại |
| | | scale_features | Chuẩn hóa các đặc trưng số |
| | | create_new_feature | Tạo đặc trưng mới |

| | | | |
|---|---|---|---|
| | | get_processed_data | Trả về DataFrame đã xử lý |
| | | new_data | Lưu dữ liệu mới ra file CSV |
| DataVisualizer | Trực quan hóa dữ liệu | _subplot | Tạo layout subplot theo số lượng cột cần vẽ |
| | | histogram | Vẽ histogram cho các cột số |
| | | barplot | Vẽ barplot cho các cột phân loại |
| | | boxplot_for_ numeric_cols | Vẽ boxplot cho từng cột số |
| | | heatmap | Vẽ heatmap ma trận tương quan |
| | | scatter | Vẽ scatter giữa từng feature số và target |
| | | heatmap_one_ column | Vẽ heatmap chỉ tương quan giữa các feature số và target |

## II. Huấn luyện mô hình

| **Tên lớp** | **Chức năng của lớp** | **Tên phương thức** | **Chức năng của phương thức** |
|---|---|---|---|
| TrainModel | Huấn luyện mô hình | _build_baseline_model | Tạo baseline cho mô hình bằng tên |
| | | fit | Fit mô hình |
| GridTuner | Tối ưu tham số bằng GridSearch | _build_scorer | Tính điểm số đánh giá cho Grid Search |
| | | optimize | Tối ưu tham số cho mô hình |

| | | | |
|---|---|---|---|
| OptunaTuner | Tối ưu tham số bằng Optuna | _evaluate | Tính điểm số đánh giá cho từng trial |
| | | optimize | Tối ưu tham số cho mô hình |
| BestModelSelector | Chọn mô hình tốt nhất | select | Lựa chọn mô hình tốt nhất và tối ưu tham số (bằng Grid hoặc Optuna) |
| EvaluateModel | Đánh giá mô hình | evaluate | Tính toán các chỉ số đánh giá |
| | | plot_confusion_matrix | Vẽ ma trận nhầm lẫn |
| | | comparison_bar_plot | Vẽ bar plot so sánh giữa các mô hình |
| | | radar_plot | Vẽ biểu đồ radar so sánh giữa các mô hình |
| SHAPExplainer | Giải thích mô hình bằng SHAP | _build_explainer | Xây dựng explainer |
| | | _compute_shap | Tính SHAP value |
| | | _select_class | Xác định lớp cần giải thích |
| | | _get_shap | Lấy SHAP value cho mẫu |
| | | inverse_row | Khôi phục giá trị đặc trưng ban đầu của mẫu để hiển thị |
| | | beeswarm | Vẽ SHAP beeswarm plot |
| | | dependence | Vẽ SHAP dependence plot |
| | | force | Vẽ SHAP force plot cho một mẫu |
| ModelTrainPipeline | Pipeline cho việc train model | load_data | Nạp dữ liệu đã xử lý |
| | | split_data | Chia tập dữ liệu thành train/test |

| | | | |
|---|---|---|---|
| | | train | Huấn luyện mô hình |
| | | optimize_params | Tối ưu tham số cho mô hình |
| | | select_best_model | Chọn mô hình tốt nhất |
| | | save_experiment_results | Lưu kết quả đánh giá |
| | | evaluate | Đánh giá mô hình và lưu kết quả |
| | | comparison_plot | Vẽ các biểu đồ so sánh cho các mô hình. |
| | | explain | Khởi tạo SHAP explainer |
| | | shap_beeswarm | Vẽ SHAP beeswarm plot |
| | | shap_dependence | Vẽ SHAP dependence plot |
| | | shap_force | Vẽ SHAP force plot cho một mẫu |
| | | save | Lưu mô hình bằng joblib |
| | | load | Nạp mô hình bằng joblib |

# C. GIỚI THIỆU TỔNG QUAN VỀ THƯ VIỆN

## I. Thư viện Scikit-learn và các mô hình máy học

### 1. Thư viện Scikit-learn

Scikit-learn là một thư viện mã nguồn mở nổi bật trong hệ sinh thái Python, đặc biệt được ưa chuộng trong lĩnh vực học máy. Thư viện được phát triển dựa trên NumPy, SciPy và Matplotlib, cung cấp một tập các công cụ xử lý các bài toán học máy và mô hình hóa thống kê gồm: phân loại, hồi quy, phân cụm và giảm chiều dữ liệu.

### 2. Mô hình Logistic Regression

Logistic Regression là một mô hình thống kê và học máy được sử dụng phổ biến để giải quyết các bài toán phân loại nhị phân. Không giống như hồi quy tuyến tính, Logistic Regression dự đoán xác suất xảy ra của một sự kiện bằng cách sử dụng *hàm sigmoid* để ánh xạ các giá trị dự đoán vào khoảng từ 0 đến 1.

### 3. Mô hình Random Forest

Random Forest (Rừng ngẫu nhiên) là một thuật toán học máy thuộc nhóm học có giám sát và được sử dụng phổ biến trong các bài toán phân loại và hồi quy. Thuật toán này là một dạng của tập hợp học, nơi mà nhiều mô hình yếu, cụ thể là các cây quyết định, được kết hợp lại để tạo thành một mô hình mạnh mẽ hơn.

- Đối với bài toán phân loại, Random Forest sẽ lấy kết quả dự đoán của từng cây và chọn kết quả nào xuất hiện nhiều nhất.

- Đối với bài toán hồi quy, kết quả cuối cùng là giá trị trung bình của các dự đoán từ tất cả các cây.

Yếu tố ngẫu nhiên đến từ hai khía cạnh: lựa chọn ngẫu nhiên các mẫu dữ liệu cũng như các đặc trưng tại mỗi lần chia nhánh trong cây.

### 4. Các siêu tham số

**** Mô hình Logistic Regression:***

- **C:** Hệ số nghịch đảo của cường độ điều chuẩn. Giá trị C càng nhỏ, mức độ điều chuẩn càng mạnh để ngăn chặn overfitting.
- **penalty:** Quy định loại điều chuẩn được áp dụng cho mô hình.
- **solver:** Thuật toán tối ưu hóa được sử dụng để tìm nghiệm của hàm mất mát.
- **max_iter:** Số lượng vòng lặp tối đa để thuật toán tối ưu hội tụ.
- **Các tham số cho nhiệm vụ học tập:**
    - **random_state:** Thiết lập seed để kiểm soát tính ngẫu nhiên, giúp kết quả mô hình tái lập được.

**** Mô hình Random Forest:***

- **n_estimators:** Số lượng cây quyết định trong rừng. Số lượng cây càng nhiều thì mô hình càng ổn định nhưng tốn thời gian huấn luyện.
- **max_depth:** Độ sâu tối đa của cây.
- **min_samples_split:** Số lượng mẫu tối thiểu cần thiết để tách một nút.
- **min_samples_leaf:** Số lượng mẫu tối thiểu cần thiết tại một nút lá.
- **n_jobs:** Số lượng luồng chạy song song khi huống luyện (-1: dùng toàn bộ CPU cores, 1: chạy tuần tự).
- **Các tham số cho nhiệm vụ học tập:**
    - **random_state:** Thiết lập seed để kiểm soát tính ngẫu nhiên, giúp kết quả mô hình tái lập được.
    - **n_jobs:** Kiểm soát số luồng chạy trong các tác vụ.

## 5. Tối ưu siêu tham số bằng GridSearchCV

Grid search là phương pháp tối ưu siêu tham số bằng cách thử tất cả các tổ hợp giá trị trong không gian tìm kiếm. Sau đó chọn ra tổ hợp có hiệu suất tốt nhất.

**Ưu điểm**: đơn giản và dễ thực hiện.

**Nhược điểm**: tốn thời gian với không gian lớn.

## 6. Ưu điểm của thư viện Scikit-learn

- **Dễ sử dụng**: Cung cấp giao diện đơn giản, thân thiện và dễ triển khai cho các tác vụ học máy.

- **Ứng dụng rộng rãi**: Tích hợp nhiều thuật toán để giải quyết các bài toán như phân loại, hồi quy, phân cụm,...

- **Công cụ tiền xử lý dữ liệu**: Cung cấp các tiện ích cho tiền xử lý, bao gồm chuẩn hóa, mã hóa dữ liệu, xử lý giá trị thiếu,...

- **Đánh giá mô hình**: Hỗ trợ nhiều chỉ số đánh giá hiệu suất (accuracy, precision, recall, F1-score), các phương pháp như đường cong ROC–AUC, kiểm định chéo (cross-validation) giúp đánh giá mô hình toàn diện và đáng tin cậy.

- **Khả năng tích hợp**: Tích hợp tốt với các thư viện Python khác như NumPy, Pandas và Matplotlib.

## II. Thư viện XGBoost

XGBoost (***Extreme Gradient Boosting***) là một thư viện máy học mã nguồn mở triển khai thuật toán *Gradient Boosting Decision Tree (GBDT)* theo cách tối ưu hóa mạnh mẽ, có khả năng mở rộng và hiệu năng cao.

### 1. Nguyên lý hoạt động

XGBoost hoạt động dựa trên nguyên lý *Ensemble Learning*:

- Xây dựng mô hình bằng cách kết hợp nhiều mô hình yếu (thường là cây quyết định).

- Các cây được thêm vào mô hình một cách tuần tự. Mỗi cây mới được huấn luyện để giảm sai số còn lại của mô hình trước đó bằng cách đi theo *gradient* của hàm mất mát — do đó có tên *Gradient Boosting*.

- Kết quả cuối cùng là tổng hợp dự đoán của tất cả các cây.

### 2. Ưu điểm

- **Hiệu năng cao:** Thường vượt trội hơn các phương pháp khác trên nhiều bộ dữ liệu.

- **Ứng dụng rộng rãi:** Dùng trong các bài toán hồi quy, phân loại.

- **Hệ sinh thái:** Sử dụng trên đa nền tảng (Windows, Linux, macOS), đa ngôn ngữ (Python, R, Java, C++) và tích hợp mạnh với AWS, Azure.

- **Sử dụng và phát triển:** Được cộng đồng sử dụng rộng rãi, tài liệu phong phú, nguồn mở liên tục được phát triển.

### 3. Các siêu tham số

- **n_estimators -** số lượng cây (*boosting rounds*): Mỗi vòng tạo một cây mới để sửa lỗi của các vòng trước, luôn kết hợp với *learning_rate*.
- **learning_rate:** Tốc độ học của mô hình, thu nhỏ đóng góp của mỗi cây được dùng trong quá trình cập nhật để ngăn *overfitting.*
- **max_depth:** Độ sâu tối đa của cây, mặc định là 6.
- **subsample:** Tỉ lệ mẫu con để train mỗi cây. Nếu đặt là 0.5 nghĩa là mô hình sẽ lấy ngẫu nhiên một nửa dữ liệu huấn luyện trước khi tạo cây → giảm *overfitting.*
- **colsample_bytree:** Tỉ lệ mẫu con của các cột khi xây dựng mỗi cây.
- **random_state:** seed để điều khiển RNG của XGBoost, giúp reproducible.
- **n_jobs:** số luồng CPU dùng cho tranning/predict.

- **verbosity:** độ chi tiết của thông báo (0 – im lặng, 1- cảnh báo, 2 – thông tin, 3 – gỡ lỗi), mặc định là 1.
- **Các tham số nhiệm vụ học tập:**
    - **objective:** *binary: logistic* – hồi quy logistic cho phân loại nhị phân trả về xác suất, *multi:softprob* – thiết lập XGBoost để thực hiện phân loại đa lớp, trả về vector xác suất cho mỗi lớp.
    - **eval_metric:** các chỉ số đánh giá cho validation data, mặc định *rmse* cho hồi quy và *logloss* cho phân loại (*logloss* cho phân loại nhị phân và *mlogloss* cho phân loại đa lớp).
    - **num_class:** đặt số lớp, bắt buộc với phân loại đa lớp.

## III. Thư viện LightGBM

LightGBM (***Light Gradient Boosting Machine***) là một thư viện mã nguồn mở được thiết kế để triển khai thuật toán *Gradient Boosting Decision Tree (GBDT)*.

### 1. Nguyên lý hoạt động

LightGBM đạt được tốc độ cao nhờ thay đổi cơ bản trong cấu trúc cây và xử lý dữ liệu:

- Khác với các thuật toán *boosting* truyền thống, LightGBM sử dụng một cách tiếp cận có tên là *Gradient-based One-Side Sampling (GOSS)* và *Exclusive Feature Bundling (EFB)* để tăng tốc quá trình huấn luyện và giảm độ phức tạp tính toán.

- LightGBM tập trung vào việc chia tách nút lá nào mang lại độ giảm hàm mất mát lớn nhất. Chiến lược này giúp mô hình hội tụ nhanh hơn và đạt độ chính xác cao hơn, nhưng cần kiểm soát chặt chẽ tham số độ sâu tối đa để tránh *overfitting*.

### 2. Ưu điểm

- **Hiệu suất cao:** LightGBM được thiết kế để đạt được hiệu suất tốt trên các bộ dữ liệu lớn, có thể huấn luyện các mô hình nhanh chóng mà không cần phải sử dụng quá nhiều bộ nhớ, giúp giảm thiểu chi phí tính toán.

- **Ứng dụng rộng rãi:** Sử dụng cho các bài toán phân loại và hồi quy.

### 3. Các siêu tham số

- **n_estimators -** số lượng cây (*boosting rounds*): Mỗi vòng tạo một cây mới để sửa lỗi của các cây trước, luôn kết hợp với *learning_rate*.
- **learning_rate:** Tốc độ học, điều khiển mức đóng góp của từng cây.
- **num_leaves:** Số lượng lá tối đa của cây (*num_leaves* nên nhỏ hơn hoặc bằng *2^(max_depth)*).

- **max_depth:** Độ sâu tối đa của cây.
- **min_child_samples:** Số lượng mẫu tối thiểu ở mỗi lá.
- **min_split_gain:** Ngưỡng gain tối thiểu để thực hiện chia tách.
- **subsample:** Tỉ lệ mẫu con để train mỗi cây.
- **colsample_bytree:** Tỉ lệ mẫu con của các cột khi xây dựng mỗi cây.
- **random_state:** seed cho reproducibility.
- **n_jobs:** số luồng CPU để trainning.
- **verbosity:** Mức độ chi tiết của thông tin hiển thị (-1 – tắt toàn bộ log).
- **Các siêu tham số cho nhiệm vụ học tập:**
  - **objective:** *binary* cho phân loại nhị phân và *multiclass* cho phân loại đa lớp.
  - **scale_pos_weight:** dùng để xử lý mất cân bằng dữ liệu (chỉ dùng cho phân loại nhị phân). Tính theo công thức *số mẫu lớp 0 / số mẫu lớp 1*.
  - **num_class:** đặt số lớp (bắt buộc cho phân loại đa lớp).

## IV. Thư viện Optuna

Optuna là một *framework* mã nguồn mở hỗ trợ việc tối ưu siêu tham số mô hình để mô hình đạt được hiệu năng tốt nhất.

### 1. Cấu trúc cơ bản

- **Objective Function:** Là một hàm mà Optuna sẽ cố gắng minimize hoặc maximize. Hàm này chứa logic huấn luyện và đánh giá mô hình.

- **Trial**: Một đối tượng đại diện cho một lần thử nghiệm với một bộ siêu tham số cụ thể. Trong *objective function*, ta sử dụng phương thức *trial.suggest* để đề xuất các giá trị cho siêu tham số.

- **Study:** Một đối tượng dùng để quản lý toàn bộ quá trình tối ưu. Nó lưu trữ lịch sử của tất cả các *Trials* và tìm ra bộ siêu tham số tốt nhất.

- **Sampler và Pruner:**

  + **Sampler:** Thuật toán được sử dụng để chọn siêu tham số tiếp theo cho *Trial*.

  + **Pruner:** Thuật toán được sử dụng để quyết định có nên dừng một *Trial* không hứa hẹn sớm hay không.

## 2. Ưu điểm

- **Không gian tìm kiếm động**: Optuna cho phép định nghĩa không gian siêu tham số một cách linh hoạt trong quá trình chạy, dựa trên các giá trị của các siêu tham số đã được chọn trước đó.

- **Hiệu suất tối ưu:** Optuna cung cấp các thuật toán lấy mẫu hiện đại giúp tìm kiếm các siêu tham số hiệu quả hơn, tập trung vào các khu vực đã cho thấy kết quả tốt, đồng thời sử dụng các Pruner để dừng sớm các Trials không hứa hẹn ngay khi chúng cho thấy hiệu suất kém.

- **Trực quan hóa và phân tích:**

    + **Công cụ trực quan hóa phong phú:** Optuna cung cấp nhiều công cụ để phân tích kết quả tối ưu hóa: *Lịch sử tối ưu hóa* (xem hiệu suất cải thiện qua từng *Trial*), *Biểu đồ quan trọng của siêu tham số* (xác định siêu tham số nào có ảnh hưởng lớn nhất đến kết quả cuối cùng), *Biểu đồ mối quan hệ* (hiển thị mối quan hệ giữa các siêu tham số và giá trị mục tiêu).

    + **Optuna Dashboard:** Cung cấp giao diện web để theo dõi tiến trình tối ưu hóa theo thời gian thực.

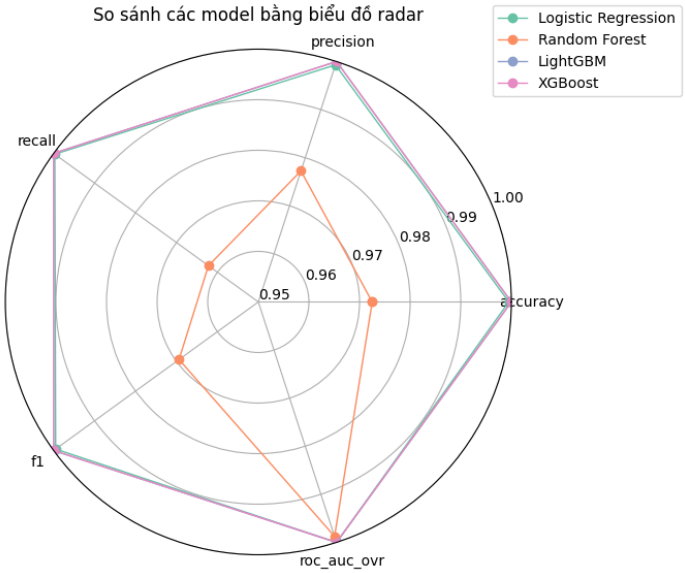
## 3. Các siêu tham số

- **Các tham số khởi tạo trong Optuna Study:**
    - **direction:** Xác định hướng tối ưu hóa của bài toán. Có hai lựa chọn là: *minimize* – tìm giá trị nhỏ nhất của *objective function* (*loss, logloss, rmse)* và *maximize* – tìm giá trị lớn nhất *(accuracy, f1, auc).*
    - **sampler:** Các sampler phổ biến là TPESampler() – tập trung sampling nhiều hơn ở các vùng có tham số cho kết quả tốt, RandomSampler() – sampling ngẫu nhiên và CmaEsSampler() – tối ưu liên tục cho không gian tham số lớn.
    - **pruner:** Các pruner phổ biến là *MedianPruner(), HyperbandPruner()* và *SuccessiveHalvingPruner()*. Tham số kèm với pruner là **n_startup_trials,** là số *trial* thực thi mà không bị pruner cắt, sau các *trial* này, pruner mới bắt đầu đánh giá và dừng sớm các trial có kết quả không tốt.
- **Tham số khi chạy Study: n_trials –** số lần mà Optuna sẽ thử với các bộ tham số khác nhau.

## D. PHÂN TÍCH KẾT QUẢ

Kết quả so sánh giữa các model:


*Biểu đồ cột so sánh các chỉ số f1, recall và ROC AUC OvR giữa các model*




*Biểu đồ radar so sánh các chỉ số giữa các model*

***Nhận xét:***

- Các mô hình đều đạt hiệu suất rất tốt trên tập dữ liệu kiểm thử.
- Hai mô hình có hiệu suất tốt nhất là LightGBM và XGBoost với các chỉ số đều là 1.0.
- Logistic Regression có các chỉ số gần như tuyệt đối.
- Random Forest có hiệu suất thấp hơn rõ rệt so với ba mô hình còn lại (các chỉ số *accuracy, precision, recall, f1* khoảng 0.96 đến 0.98).

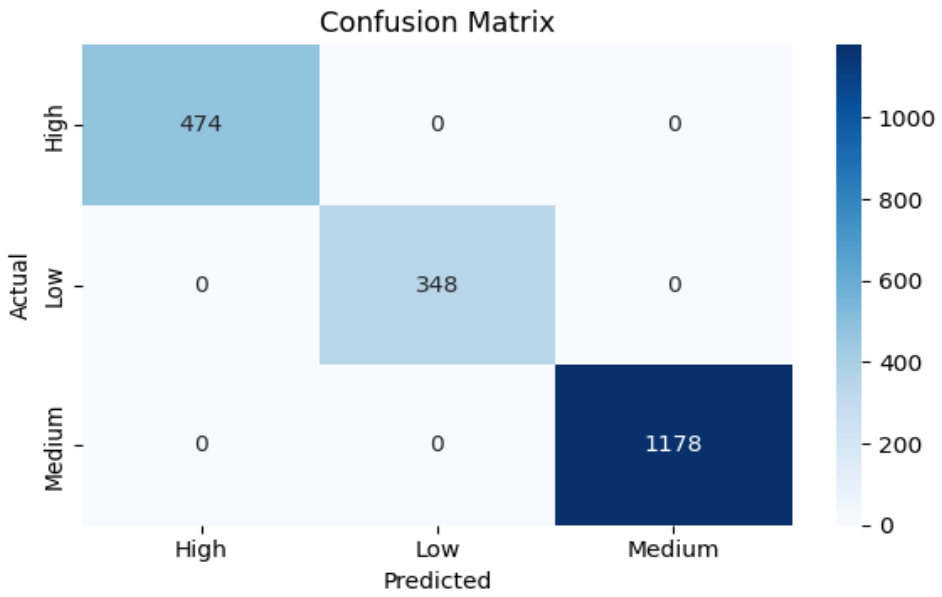
⇒ Các mô hình đều có kết quả rất tốt. XGBoost và LightGBM thể hiện hiệu suất rất mạnh mẽ, Logistic Regression cũng đạt điểm tối đa cho thấy dữ liệu có thể tuyến tính hoặc có tính phân tách cao, Random Forest hoạt động tốt nhưng không tối ưu bằng các mô hình khác.

Khi chọn mô hình tốt nhất, kết quả đánh giá là như nhau đối với LightGBM và XGBoost nên nhóm sẽ phân tích kết quả của hai mô hình này.

## I. Phân tích kết quả dựa trên mô hình XGBoost (đã tối ưu tham số)

### 1. Nhận xét kết quả

| accuracy | precision | recall | f1 | roc_auc | roc_auc_ovr |
|---|---|---|---|---|---|
| 1.0 | 1.0 | 1.0 | 1.0 | None | 1.0 |




*Kết quả khi train model: (như nhau đối với cả 2 cách tối ưu)*

- Các chỉ số hiệu suất đều là 1.0 cho thấy rằng mô hình được huấn luyện có hiệu suất rất tốt trên tập dữ liệu kiểm thử.

- Phân tích ma trận nhầm lẫn:

  + Lớp *High:* Mô hình đã dự đoán chính xác 474 mẫu thực tế.

  + Lớp *Low:* Mô hình đã dự đoán chính xác 348 mẫu thực tế.

  + Lớp *Medium:* Mô hình đã dự đoán chính xác 1178 mẫu thực tế.

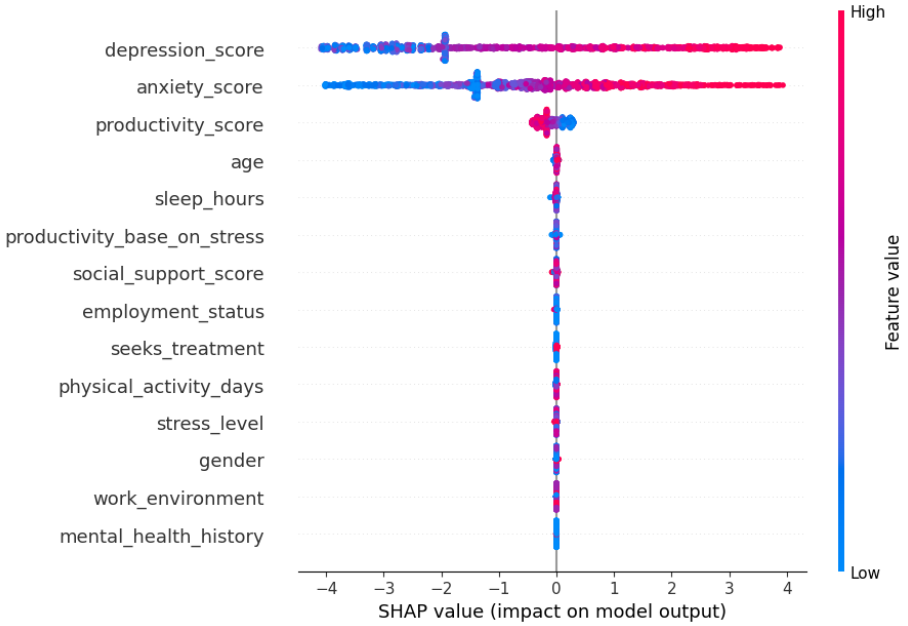
⇒ Mô hình đã dự đoán đúng tất cả các mẫu trên tổng số 2000 mẫu của tập kiểm thử và không có bất kỳ dự đoán nhầm lẫn nào.

***Nhận xét:***

- Mô hình đã phân loại rất tốt các mẫu kiểm thử cho thấy tập dữ liệu này có thể rất dễ phân loại (dữ liệu quá sạch và không nhiễu).

- Kết quả dự đoán chính xác 100% có thể là dấu hiệu của việc dữ liệu kiểm thử bị rò rỉ vào tập huấn luyện khiến mô hình đạt hiệu suất không thực tế.

## 2. Giải thích kết quả mô hình bằng SHAP




### *a. Beeswarm plot*

Biểu đồ SHAP Beeswarm cho thấy tầm quan trọng của các đặc trưng và hướng tác động của chúng lên kết quả đầu ra của mô hình. Do đó dựa vào biểu đồ này, chúng ta có một số nhận xét sau:

- Ba đặc trưng có ảnh hưởng lớn nhất đến kết quả mô hình là *depression_score, anxiety_score và productivity_score:*

  + Điểm trầm cảm (*depression_score):* Đây là đặc trưng quan trọng nhất. Các chấm đỏ (điểm trầm cảm cao) nằm chủ yếu ở phía bên phải, các chấm xanh dương (điểm trầm cảm thấp) tập trung ở phía bên trái và các chấm trải dài cho thấy điểm trầm cảm có tác động lớn đến kết quả đầu ra của mô hình. Nói cách khác, giá trị *depression_score* cao có xu hướng đẩy đầu ra mô hình theo chiều tăng, trong khi giá trị thấp có xu hướng làm giảm đầu ra.

  + Điểm lo âu (*anxiety_score)*: Đây là đặc trưng quan trọng thứ hai. Tương tự với điểm trầm cảm, đặc trưng này cũng có tương quan thuận với kết quả đầu ra của mô hình. Điểm lo âu cao làm tăng kết quả đầu ra và điểm lo âu thấp làm giảm kết quả đầu ra của mô hình.

  + Điểm năng suất (*productivity_score)*: Đây là đặc trưng quan trọng thứ ba và có mối tương quan ngược lại so với hai đặc trưng trên. Các chấm màu đỏ chủ yếu nằm phía bên trái, các chấm màu xanh dương chủ yếu nằm ở phía bên phải và các chấm tập trung chủ yếu ở trong khoảng SHAP value từ -1

đến 1 cho thấy mối tương quan nghịch khá yếu giữa điểm năng suất và kết quả mô hình.

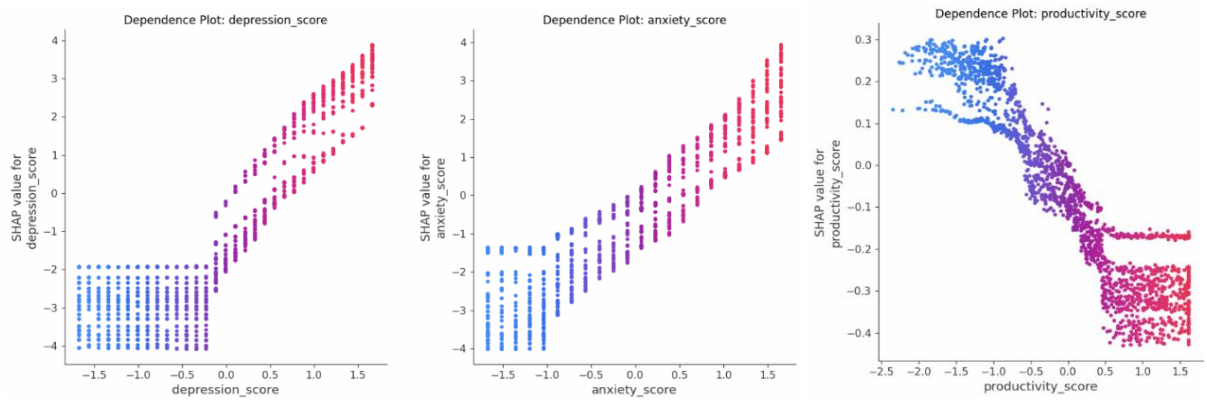
- Các đặc trưng còn lại hầu như không có đóng góp đáng kể đến kết quả đầu ra của mô hình (các chấm chủ yếu tập trung ở SHAP value bằng 0).

**Kết luận**

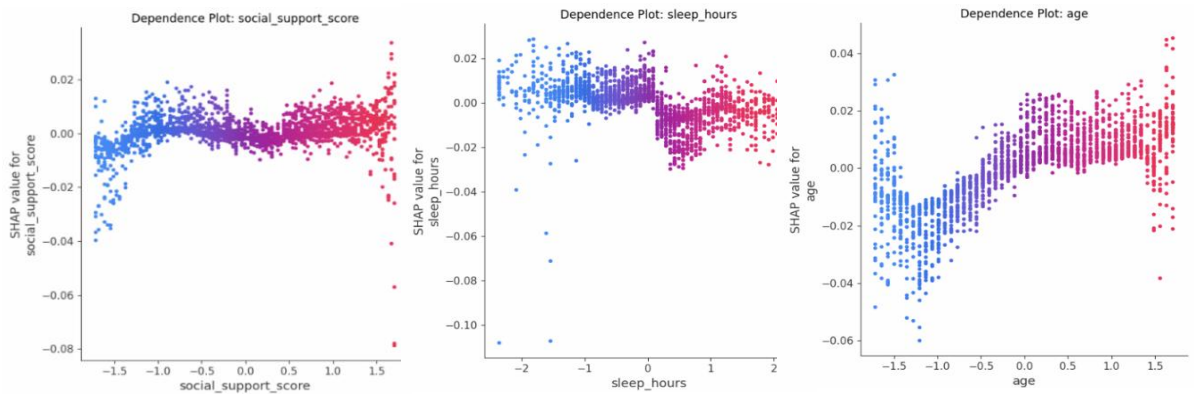
Phân tích beeswarm cho thấy các chỉ số tâm lý như *depression_score* và *anxiety_score* là những đặc trưng chi phối chính đầu ra của mô hình theo chiều tăng, trong khi *productivity_score* thể hiện mối quan hệ nghịch với SHAP value. Các đặc trưng còn lại có vai trò hạn chế trong việc ảnh hưởng đến đầu ra của mô hình.

***b. Dependence plot***

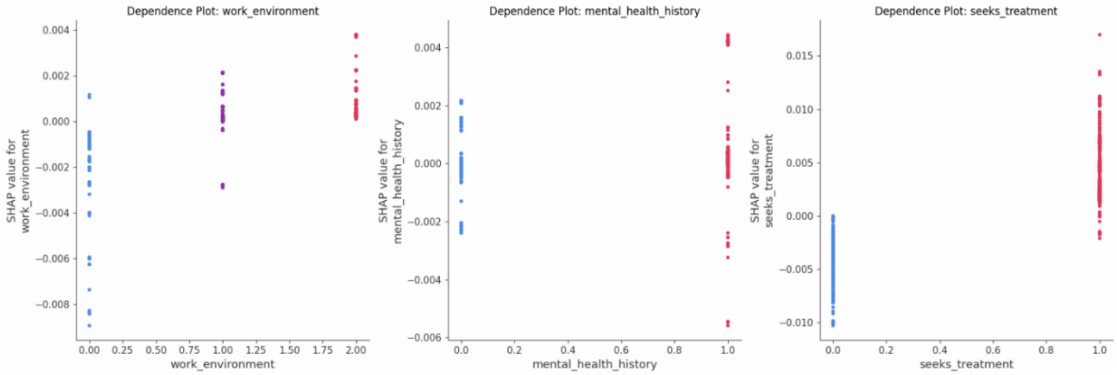
- Các đặc trưng *depression_score* và *anxiety_score* đều cho thấy quan hệ tuyến tính thuận với SHAP value. Trong khi đó *productivity_score* cho thấy quan hệ tuyến tính nghịch với SHAP value.




- Một số đặc trưng như *social_support_score, sleep_hours và age* có quan hệ phi tuyến phức tạp hơn với SHAP value.



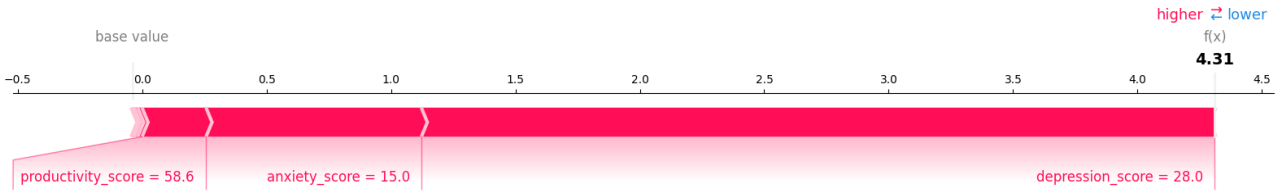
- Hầu hết các đặc trưng còn lại có SHAP value rất thấp (khoảng gần 0).




## Kết luận

Thông qua *dependence plot,* kết quả mô hình bị chi phối nhiều bởi các chỉ số tâm lý và năng suất làm việc. Các tác động từ môi trường và yếu tố nhân khẩu học không có ảnh hưởng đáng kể đến việc dự đoán.

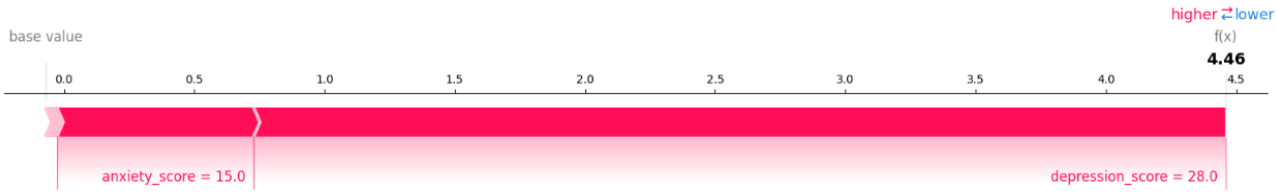
### *c. Force plot (cho mẫu 0)*




*(Kết quả của mô hình tối ưu tham số bằng Optuna)*

Biểu đồ force minh họa đóng góp của các đặc trưng riêng lẻ vào việc chuyển dự đoán từ *base value* (đầu ra trung bình của mô hình trên toàn bộ tập huấn luyện) sang giá trị đầu ra thô *f(x)* cho một mẫu cụ thể.

- Base value khoảng 0 - đây là giá trị khởi đầu cho việc dự đoán.
- Giá trị *f(x)* = 4.31 cao hơn đáng kể so với base value, cho thấy các đặc trưng của mẫu đang đóng góp mạnh theo hướng làm tăng đầu ra của mô hình cho lớp đang xét.
- *depression_score* = 28.0 là đặc trưng có đóng góp lớn nhất theo hướng làm tăng giá trị đầu ra của mô hình.
- *anxiety_score* = 15.0 là đặc trưng làm tăng giá trị đầu ra lên mạnh thứ hai.
- *productivity_score* = 58.6 làm tăng nhẹ giá trị đầu ra của mô hình.




*(Kết quả của mô hình tối ưu tham số bằng GridSearch)*

- Tương tự như biểu đồ force cho mô hình tối ưu bằng Optuna, *depression_score* và *anxiety_score* vẫn là hai đặc trưng có tác động đẩy giá trị dự đoán của mô hình tăng lên (khoảng 4.46).

**Kết luận**

Giá trị đầu ra của mô hình cho mẫu này bị đẩy lên đáng kể do đóng góp mạnh của các chỉ số tâm lý. Bên cạnh đó điểm năng suất cũng góp phần nhẹ trong việc tăng giá trị đầu ra của mô hình.

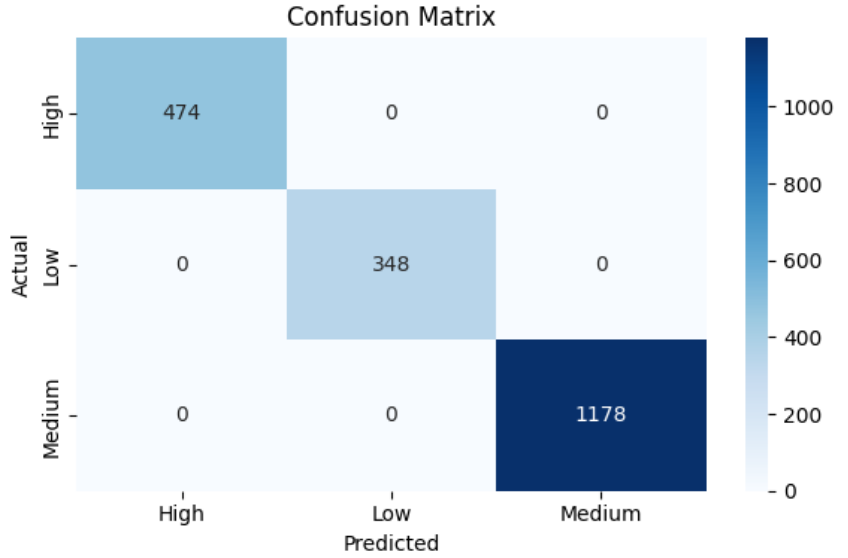
***Nhận xét:***

- Mô hình đạt hiệu suất phân loại rất cao trên tập kiểm thử. Tuy nhiên, kết quả này cần được xem xét một cách thận trọng do khả năng bị rò rỉ dữ liệu hoặc do đặc tính của bộ dữ liệu khiến bài toán trở nên dễ phân loại.
- Phân tích SHAP cho thấy đầu ra của mô hình chịu chi phối chủ yếu bởi các chỉ số sức khỏe tâm lý như *depression_score* và *anxiety_score*, trong khi các yếu tố nhân khẩu học và môi trường có mức đóng góp không đáng kể.
- Đối với một mẫu cụ thể, các chỉ số tâm lý cao có thể đóng vai trò chi phối mạnh, làm tăng giá trị đầu ra của mô hình cho lớp đang được xét.

## II. Phân tích kết quả dựa trên mô hình LightGBM (đã tối ưu tham số)

### 1. Nhận xét kết quả

| | accuracy | precision | recall | f1 | roc_auc | roc_auc_ovr |
|---|---|---|---|---|---|---|
| 0 | 1.0 | 1.0 | 1.0 | 1.0 | None | 1.0 |




*Kết quả khi train model: (như nhau đối với cả 2 cách tối ưu)*

- Tương tự như XGBoost, LightGBM đạt các chỉ số hiệu suất (accuracy, precision, recall, f1 và roc_auc_ovr) đều ở mức tối đa là 1.0 trên tập dữ liệu kiểm thử.

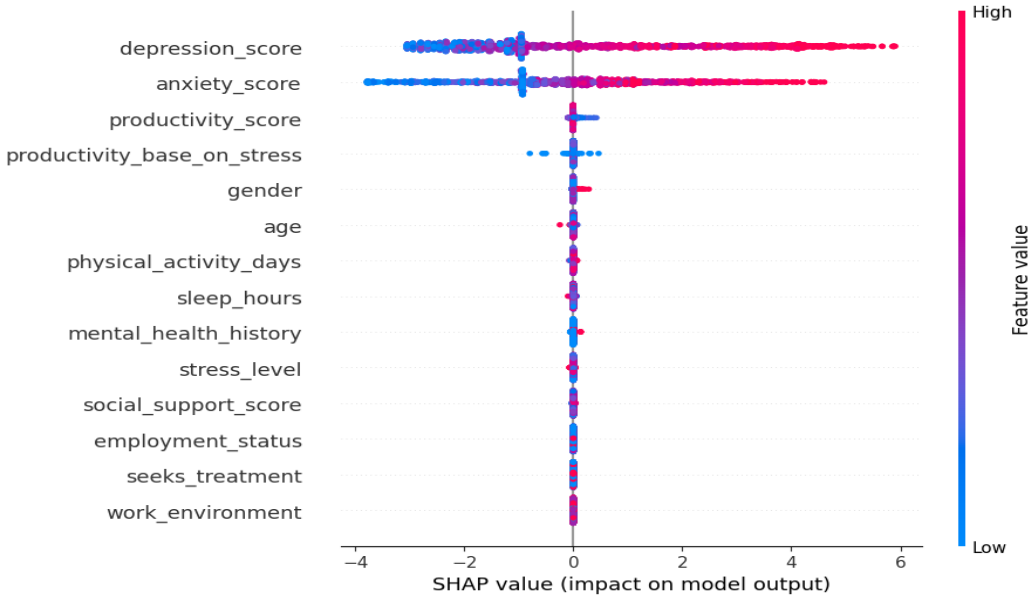
- Ma trận nhầm lẫn cho thấy sự phân loại chính xác các lớp *High, Low* và *Medium*.

**Nhận xét:**

Mô hình có hiệu suất rất tốt. Tuy nhiên, việc không có bất kỳ sai số nào (độ chính xác 100%) tiếp tục là một tín hiệu cảnh báo về khả năng dữ liệu quá "sạch" (thiếu tính thực tế) hoặc hiện tượng rò rỉ dữ liệu giữa tập huấn luyện và kiểm thử .

## 2. Giải thích mô hình bằng biểu đồ SHAP

### *a. Beeswarm plot*




*(Kết quả của mô hình tối ưu tham số bằng GridSearch)*

- Biểu đồ SHAP Beeswarm của LightGBM cho thấy sự tương đồng lớn với XGBoost về mức độ quan trọng của các đặc trưng.

- Ba đặc trưng có tác động mạnh mẽ nhất đến quyết định của mô hình vẫn là *depression_score, anxiety_score* và *productivity_score*.

    + *depression_score* và *anxiety_score*: Hai đặc trưng này có dải phân bố rộng nhất trên trục hoành của biểu đồ, khẳng định đây là hai yếu tố quan trọng, đóng góp lớn nhất đến kết quả dự báo của mô hình. Cả hai đều có mối tương quan thuận rõ rệt với kết quả của mô hình.

    + *productivity_score*: Đặc trưng này thể hiện mối tương quan nghịch với SHAP value.

- Các đặc trưng khác: Đặc trưng xếp thứ 4 là *productivity_base_on_stress* có tác động yếu nhưng vẫn ghi nhận sự phân tách nhất định. Các đặc trưng còn lại có các chấm chủ yếu tập trung ở SHAP value bằng 0, hầu như không có đóng góp đáng kể đến kết quả của mô hình.

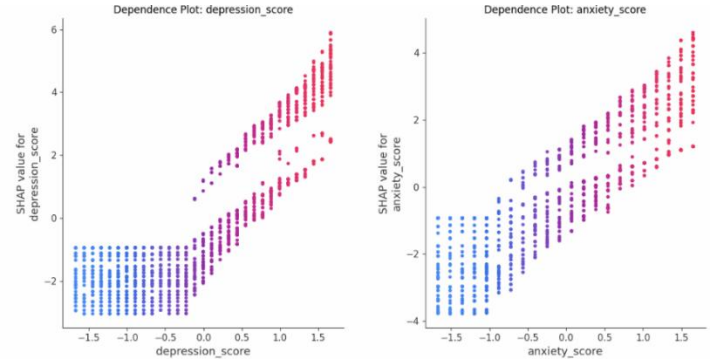
**Kết luận**

- Beeswarm plot cho thấy các đặc trưng *depression_score* và *anxiety_score* có SHAP value lớn nhất và các giá trị cao của hai đặc trưng này chủ yếu đóng góp theo chiều làm tăng đầu ra của mô hình.

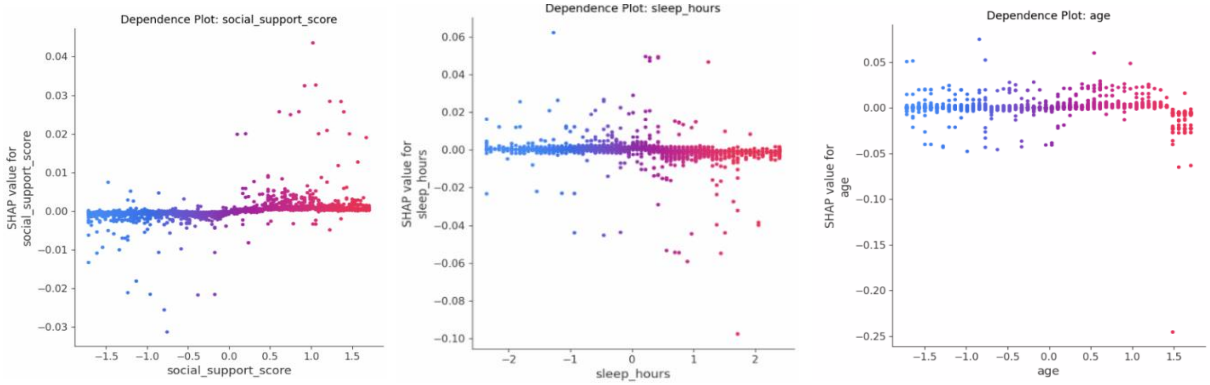
- Đối với *productivity_score*, beeswarm plot thể hiện mối quan hệ nghịch giữa giá trị của đặc trưng và SHAP value, tức là các giá trị năng suất cao thường gắn với SHAP value thấp hơn.

### *b. Dependence plot*

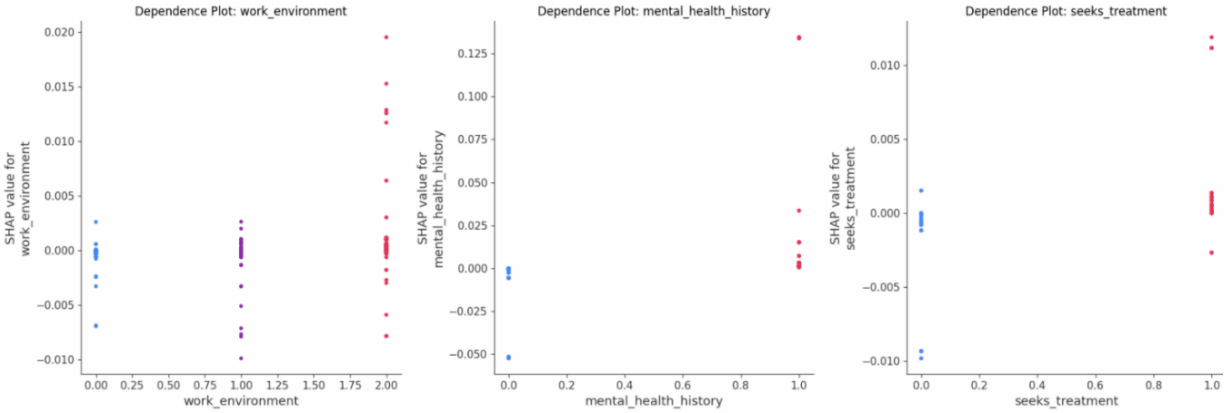
- Các đặc trưng *depression_score* và *anxiety_score* đều cho thấy quan hệ tuyến tính thuận rất rõ ràng với SHAP value.




- Một số đặc trưng như *social_support_score*, *sleep_hours* và *age* có sự phân bố phức tạp hơn (phi tuyến) nhưng biên độ dao động của SHAP value khá nhỏ so với nhóm đặc trưng chính.



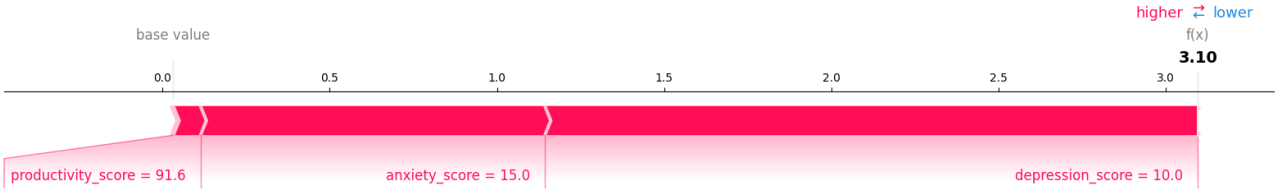

- Hầu hết các đặc trưng còn lại có SHAP value rất thấp (các điểm dữ liệu nằm trải dài trên đường thẳng sát trục 0).



**Kết luận**

Thông qua *dependence plot*, kết quả dự báo của LightGBM bị chi phối gần như hoàn toàn bởi các chỉ số sức khỏe tâm lý. Các tác động từ môi trường làm việc và yếu tố nhân khẩu học không có ảnh hưởng đáng kể đến việc dự đoán của mô hình.

### *c. Force plot (cho mẫu 0)*




*(Kết quả của mô hình tối ưu tham số bằng GridSearch)*

- Kết quả dự đoán *f(x)*: Mô hình LightGBM đưa ra giá trị dự đoán là 3.10. Giá trị này cao hơn so với giá trị cơ sở (base value), cho thấy các đặc trưng của mẫu có xu hướng đẩy đầu ra của mô hình theo chiều tăng cho lớp đang được dự đoán.

- Phân tích tác động: *depression_score = 10.0, anxiety_score = 15.0* và *productivity_score = 91.6* là các đặc trưng có tác động làm tăng giá trị đầu ra của mô hình.

**Kết luận:** *depression_score* và *anxiety_score* vẫn là hai yếu tố làm tăng giá trị đầu ra của mô hình. Bên cạnh đó *productivity_score* cũng có đóng góp nhẹ vào kết quả.

**TỔNG KẾT CHUNG:**

- Phân tích SHAP cho thấy các mô hình chủ yếu dựa vào các chỉ số tâm lý như *depression_score* và *anxiety_score* trong việc đưa ra quyết định phân loại, cho thấy các đặc trưng này có mức đóng góp lớn vào đầu ra của mô hình.
- Mô hình LightGBM cho kết quả nhất quán với XGBoost trong việc phân loại mẫu này theo cùng một chiều, phản ánh tính ổn định của các quyết định mô hình khi sử dụng tập đặc trưng hiện tại.
- Cả hai mô hình XGBoost và LightGBM đều thể hiện khả năng nắm bắt các tương tác cục bộ giữa các đặc trưng, cung cấp góc nhìn chi tiết hơn ở mức độ cá nhân so với các phân tích chỉ dựa trên từng biến riêng lẻ.

- Các biểu đồ SHAP cho thấy rằng *depression_score* và *anxiety_score* có quan hệ gần như tuyến tính với SHAP value cũng như chi phối phần lớn kết quả đầu ra của mô hình. Mặc khác, bộ dữ liệu được tạo nên bằng cách tổng hợp/mô phỏng lại các mô hình, phân bố và xu hướng được quan sát trong các tập dữ liệu và khảo sát sức khỏe tinh thần nên rất có thể *target* được xây dựng một cách trực tiếp hoặc gián tiếp từ hai đặc trưng quan trọng trên. Đây có thể là nguyên nhân chính dẫn đến hiệu suất phân loại cao bất thường của các mô hình.

## VAI TRÒ CỦA TỪNG THÀNH VIÊN

| Họ và tên | Vai trò |
|---|---|
| Nguyễn Thị Kim Hằng | Thiết kế lớp mô hình máy học, viết báo cáo |
| Nguyễn Thị Xuân Dung | Thiết kế lớp mô hình máy học, viết báo cáo |
| Nguyễn Thị Khánh Ly | Xây dựng lớp tiền xử lý, tạo file README.md và requirements.txt |

*Các phần còn lại do tất cả thành viên đóng góp.*