Exploratory Data Analysis (EDA) & SVM Modeling Report

Dataset Overview

- **Dataset Name:** `dataset.csv`
- **Target Variable:** `Disease`
- **Features:** Symptom-based (e.g., `Symptom_1` to `Symptom_N`)
- **Preprocessing Applied:** Label encoding, binary encoding, standard scaling
- **Objective:** Classify diseases based on reported symptoms

---

Target Variable

The `Disease` column is a **multi-class categorical** variable, encoded numerically using `LabelEncoder`.

> A bar plot (not shown here) should be used to visualize class distribution and detect imbalance.

Feature Insights

- The feature columns represent symptoms with categorical string values.
- Many symptom values are missing (`NaN` or blank).
- All symptom values were unified into a unique indexed list before being **binary encoded** for dimensionality reduction.

⚙ Data Preprocessing Pipeline

1. **Symptom Encoding**: Mapped all unique symptoms to integers.
2. **Binary Encoding**: Converted symptom indexes into binary to reduce dimensionality.
3. **Feature Scaling**: Used `StandardScaler` to normalize values.
4. **Data Split**: 80/20 train-test split using `train_test_split`.

Feature Selection

Used a **Random Forest Classifier** to identify and retain only the most important features via `SelectFromModel`.

> Helped improve SVM performance by removing noise.

SVM Modeling

- **Classifier Used:** Support Vector Machine (`SVC`)
- **Kernel:** RBF (Radial Basis Function)
- **Hyperparameter Tuning:** `GridSearchCV`
- **Scoring Metric:** Accuracy

Results: - Best Cross-validation Accuracy: `0.9929` - Tuned Model Accuracy (Test Set): `0.9919`

Evaluation

Used `classification_report()` for detailed evaluation. Output was visualized using a **heatmap** for precision, recall, and F1-score per class.

- **Number of Classes:** 41
- **Visualization Enhancements:**
  - Small font for class names
  - Padding and larger figure size for clarity
  - Colored summary row for average scores

(Include this visualization in your report or slide deck.)

Explainability (Optional)

- `shap` was imported, though not applied in the notebook.
- SHAP can be used for model interpretation and feature contribution explanation if needed.

Summary

- Binary encoding handled high-cardinality categorical features effectively.
- Random Forest feature selection improved model focus.
- SVM with hyperparameter tuning achieved nearly perfect classification.
- Class-wise metrics were visualized to improve interpretability.

Recommendations

- Add `df.info()` and `df.describe()` summaries to show structure and distribution.
- Visualize class distribution and missing data using seaborn/matplotlib.
- Export SHAP visualizations if interpretability is needed.