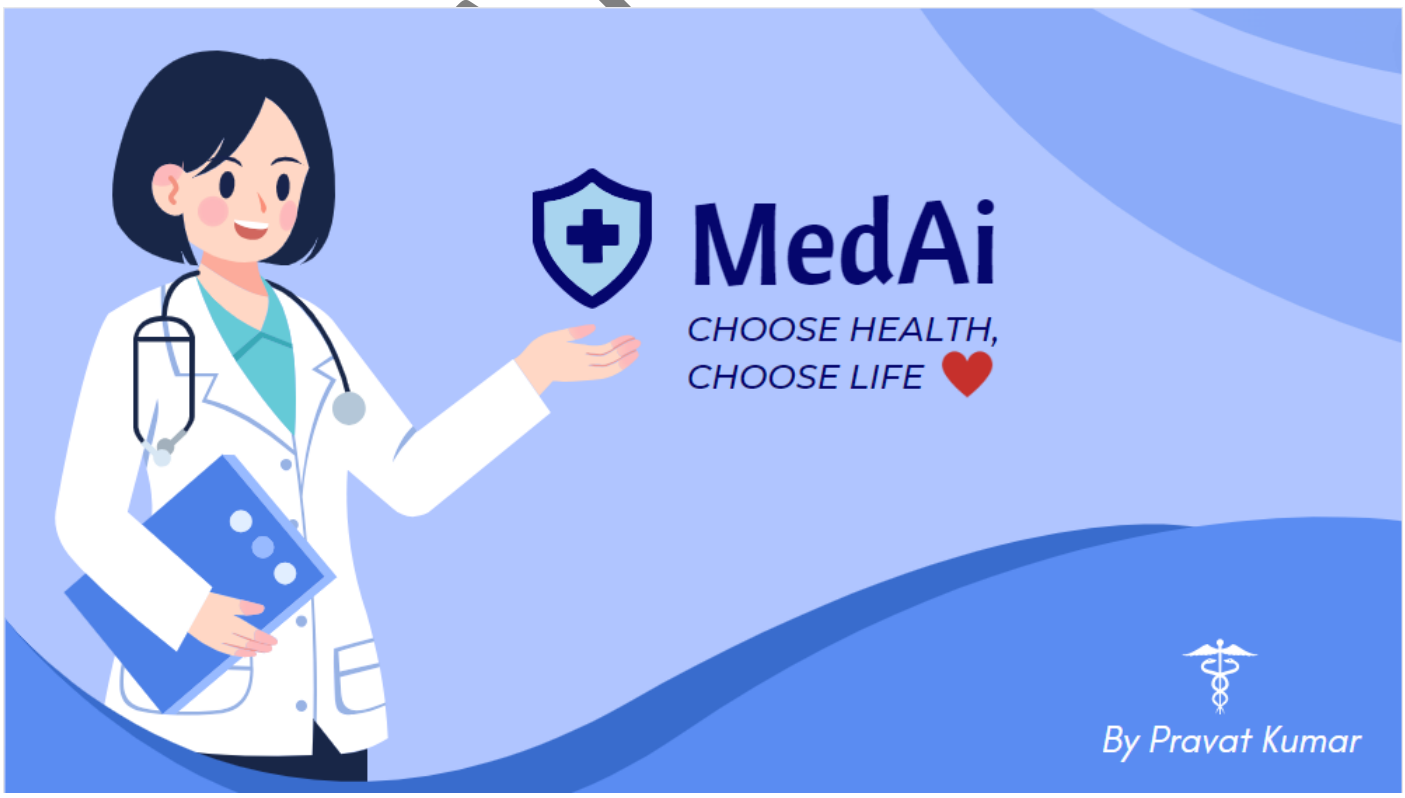
Disease Prediction Through Symptoms

Using Machine Learning

By- Pravat Kumar Panda

Date- 2nd June, 2023



Abstract:

The Disease Prediction project aims to develop a machine learning-based system for predicting diseases based on a given set of symptoms. The project utilizes a dataset containing symptom and disease information to train and evaluate different classification models. The dataset is preprocessed to handle missing values and ensure data integrity.

Three models, namely Support Vector Machine (SVM), Gaussian Naive Bayes, and Random Forest, are implemented and evaluated using cross-validation techniques. The models are trained on a portion of the dataset and tested on both the training and testing sets. Performance metrics, such as accuracy and confusion matrices, are calculated to assess the models' predictive capabilities.

Furthermore, the project deploys the best-performing models to create a combined model that takes input symptoms and predicts the most probable disease. The combined model incorporates the predictions from all three models to achieve a more robust and accurate prediction.

The system provides an interface where users can input their symptoms, and the machine learning model will generate predictions based on the trained models. The predictions can assist users in getting preliminary insights into potential diseases and seeking appropriate medical advice. It is essential to note that the system does not provide medical advice but serves as an informational tool for initial disease prediction.

Overall, the Disease Prediction project demonstrates the application of machine learning algorithms in healthcare to assist in disease prediction. By leveraging symptom data and classification models, the system offers a valuable resource for early disease identification and supports informed decision-making in seeking medical attention.

Problem Statement:

The healthcare sector faces numerous challenges in accurately and efficiently diagnosing diseases, especially when dealing with a large number of patients with diverse symptoms. Manual diagnosis based on symptoms alone can be time-consuming, prone to human errors, and may not leverage the full potential of available data. Therefore, there is a need for an automated system that can predict diseases based on symptoms, aiding healthcare professionals in making timely and accurate diagnoses.

The problem addressed by this project is the development of a disease prediction system using machine learning techniques. The system aims to analyze a given set of symptoms and provide predictions of the most probable diseases. By leveraging a dataset containing symptom and disease information, the project aims to train and evaluate various machine learning models for disease prediction. The challenge lies in selecting the most appropriate models, preprocessing the data to handle missing values, ensuring data integrity, and achieving high prediction accuracy.

The successful implementation of this project will provide healthcare professionals with a tool to support their decision-making process, allowing for faster and more accurate disease diagnosis. It will empower individuals to gain preliminary insights into potential diseases based on their symptoms and seek appropriate medical advice promptly.

Objective:

The objective of this project is to develop a disease prediction system using machine learning techniques. The project aims to build accurate models that can predict diseases based on input symptoms. The system will provide a user-friendly interface for users to input their symptoms and receive predictions of potential diseases. The models will be trained and evaluated using appropriate metrics to ensure their effectiveness. The ultimate goal is to create a reliable and efficient tool for disease prediction, enabling early diagnosis and prompt medical interventions.

Market / Customer / Business Need Assessment:

- **Market Need Assessment:**

There is a strong demand for a user-friendly and accurate disease prediction system in the healthcare industry. Individuals seek convenient ways to assess their health conditions based on symptoms, while healthcare professionals need reliable tools for diagnosis and treatment planning.

- **Customer Need Assessment:**

Customers require a reliable, accessible, and user-friendly disease prediction system. Individuals value accuracy, promptness, and ease of use to gain insights into their health conditions. Healthcare professionals need an efficient tool to support their diagnostic decision-making process and enhance patient care.

- **Business Need Assessment:**

Developing a comprehensive disease prediction system presents significant business opportunities. There is a growing market for health-related technologies, and a well-designed system can attract a large customer base. The system can create revenue streams through subscription-based services, partnerships, or licensing, establishing a strong brand presence in the healthcare technology sector.

Target Specifications and Characterizations:

The target users for this disease prediction system can include:

- **Individuals**: People who are experiencing symptoms and want to gain initial insights into possible diseases based on their symptoms. They can use the system as a self-assessment tool before seeking professional medical advice.
- **Healthcare Professionals**: Doctors, nurses, and other healthcare professionals can utilize the system as a complementary tool during the diagnostic process. It can provide additional information and suggestions for consideration, aiding in the decision-making process.
- **Medical Students and Researchers**: Students studying medicine and researchers in the healthcare field can use the system for educational and research purposes. It can serve as a reference tool to understand the relationship between symptoms and diseases.

- **Telemedicine and Remote Healthcare Services**: In the context of telemedicine or remote healthcare services, the system can assist healthcare providers in remote consultations by providing preliminary disease predictions based on reported symptoms.

- **Health Awareness Organizations:** Organizations focused on promoting health awareness and education can utilize the system to provide valuable information and resources to the general public. It can contribute to spreading knowledge about various diseases and their symptoms.

Characterizations:

- **Accuracy**: The disease prediction system should strive for a high level of accuracy in predicting diseases based on given symptoms. It should aim to minimize false positives and false negatives to provide reliable results to users.
- **User-Friendly Interface**: The system should have a user-friendly interface that is easy to navigate and understand. It should provide clear instructions for inputting symptoms and display the predicted diseases in a clear and concise manner.
- **Scalability**: The system should be designed to handle a large volume of users and scale effectively to accommodate increasing demand. It should be able to handle simultaneous requests and maintain a high level of performance and responsiveness.
- **Privacy and Security**: The system should prioritize the privacy and security of user data. It should implement robust security measures to protect sensitive user information and comply with relevant data protection regulations.
- **Integration Capability**: The system should have the capability to integrate with other healthcare systems and databases, allowing seamless exchange of information and facilitating collaboration among healthcare professionals.
- **Continuous Improvement**: The system should have provisions for continuous improvement and updates. It should incorporate feedback from users and healthcare professionals to enhance its prediction accuracy and usability over time.
- **Compatibility**: The system should be compatible with multiple platforms and devices, including web browsers, mobile devices, and tablets, to ensure accessibility for a wide range of users.
- **Speed and Performance**: The system should provide fast and efficient predictions, ensuring quick response times to user queries. It should be able to handle large datasets efficiently to deliver timely results.
- **Robustness**: The system should be robust and able to handle variations in input data, including missing or incomplete symptoms. It should have mechanisms to handle uncertainties and make predictions based on available information.

External Search:

Reference:

- https://www.geeksforgeeks.org/disease-prediction-using-machine-learning/

Dataset:

- https://www.kaggle.com/neelima98/disease-prediction-using-machine-learning

Research Papers:

- https://www.ijraset.com/research-paper/medical-disease-prediction-using-ml-algorithms
- https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3661426

Bench marking alternate products:

When benchmarking alternate products for the disease prediction project, it is essential to evaluate existing systems or tools that offer similar functionality. Here are a few potential benchmarking candidates:

- **WebMD Symptom Checker**: WebMD provides an online symptom checker that allows users to input their symptoms and receive a list of possible conditions. It serves as a widely recognized and popular platform in the field of symptom-based disease prediction.
- **Mayo Clinic Symptom Checker**: Mayo Clinic offers an online symptom checker tool that helps users assess their symptoms and provides information on potential causes and next steps. It is known for its reliable medical content and trusted reputation.
- **Ada Health**: Ada Health is a mobile app that uses artificial intelligence to provide personalized health assessments. It offers symptom analysis and generates potential diagnoses based on user inputs.
- **Isabel Healthcare**: Isabel Healthcare is a diagnostic decision support tool used by healthcare professionals. It enables doctors to input patient symptoms and medical history to receive a list of potential diagnoses.

Our disease prediction project offers several advantages that make it stand out from existing products:

- **Machine Learning-Based Approach**: Our project utilizes machine learning algorithms to improve the accuracy and reliability of disease prediction. By training the model on a comprehensive dataset, it can learn patterns and make more accurate predictions compared to rule-based systems used by other products.
- **Personalized Recommendations**: Our project takes into account the individual's specific symptoms and medical history to generate personalized disease predictions. This personalized approach enhances the accuracy and relevance of the results, providing users with more tailored insights.
- **Integration with Django Framework**: Our project is built on the Django framework, allowing for seamless integration with existing healthcare systems or platforms. This enables healthcare providers to incorporate the disease prediction functionality into their workflows, enhancing the overall efficiency of the healthcare process.
- **Open Source and Customizable**: Our project is open source, allowing users to customize and extend its functionality to meet specific requirements. This flexibility empowers healthcare professionals and developers to adapt the system to their unique needs and make improvements over time.

Business Model: Disease Prediction Application

Cost Structure:

- **Development and Maintenance Costs**: This includes the cost of developing and updating the application, ensuring its compatibility with different platforms, and maintaining the infrastructure.
- **Data Acquisition Costs**: Acquiring and maintaining a comprehensive and reliable dataset for training the machine learning models involves expenses such as data licensing or collection.
- **Hosting and Infrastructure Costs**: The application needs servers, databases, and other infrastructure components to run smoothly, which incur ongoing hosting and maintenance expenses.
- **Marketing and Promotion Costs**: To reach the target users, marketing and promotion activities, such as online advertising, content creation, and social media campaigns, are necessary.

Revenue Model:

- **Freemium Model**: Offer the basic features of the disease prediction application for free to attract a wide user base. Additional premium features, such as advanced analytics, personalized recommendations, or integration with electronic health records, can be offered at a subscription fee.
- **Data Licensing**: Collaborate with healthcare institutions, research organizations, or pharmaceutical companies to license anonymized and aggregated user data for research purposes or to enhance the accuracy of the disease prediction models.
- **Partnerships and Integration**: Establish partnerships with healthcare providers or platforms to integrate the disease prediction application into their existing systems. Revenue can be generated through licensing fees or revenue-sharing agreements.
- **Consultation Services**: Offer consultation services to healthcare professionals, institutions, or researchers seeking expertise in machine learning-based disease prediction and analytics.

By adopting a combination of these revenue streams, the business can generate sustainable revenue while providing valuable services to its users and partners.

Applicable Constraint

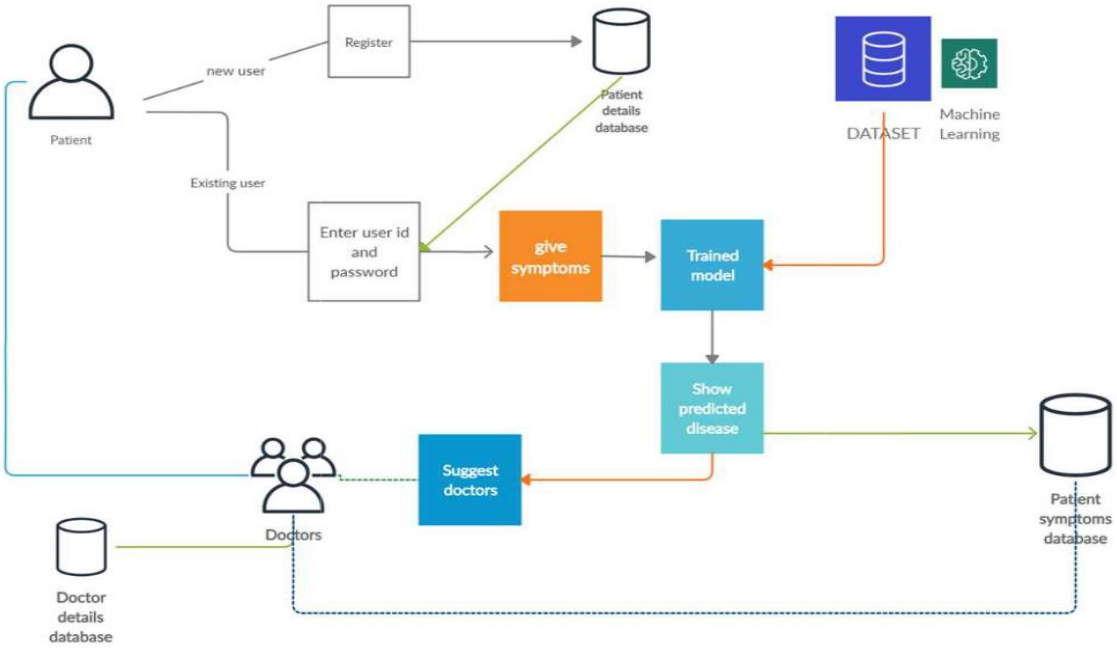
- **Data Privacy and Security**: The project must adhere to strict data privacy regulations to ensure the protection and confidentiality of user information. Measures such as encryption, secure storage, and access controls should be implemented.
- **Scalability and Performance**: The application should be able to handle a large number of users and perform efficiently, even during peak usage periods. It should be designed to scale seamlessly as the user base grows, ensuring a smooth user experience.
- **Regulatory Compliance**: The project needs to comply with applicable healthcare and data protection regulations, such as HIPAA (Health Insurance Portability and Accountability Act) in the United States or GDPR (General Data Protection Regulation) in the European Union.

- **Availability and Reliability**: The application should have high availability and reliability, minimizing downtime and ensuring uninterrupted access to users. Robust backup and disaster recovery mechanisms should be in place to handle any potential failures.

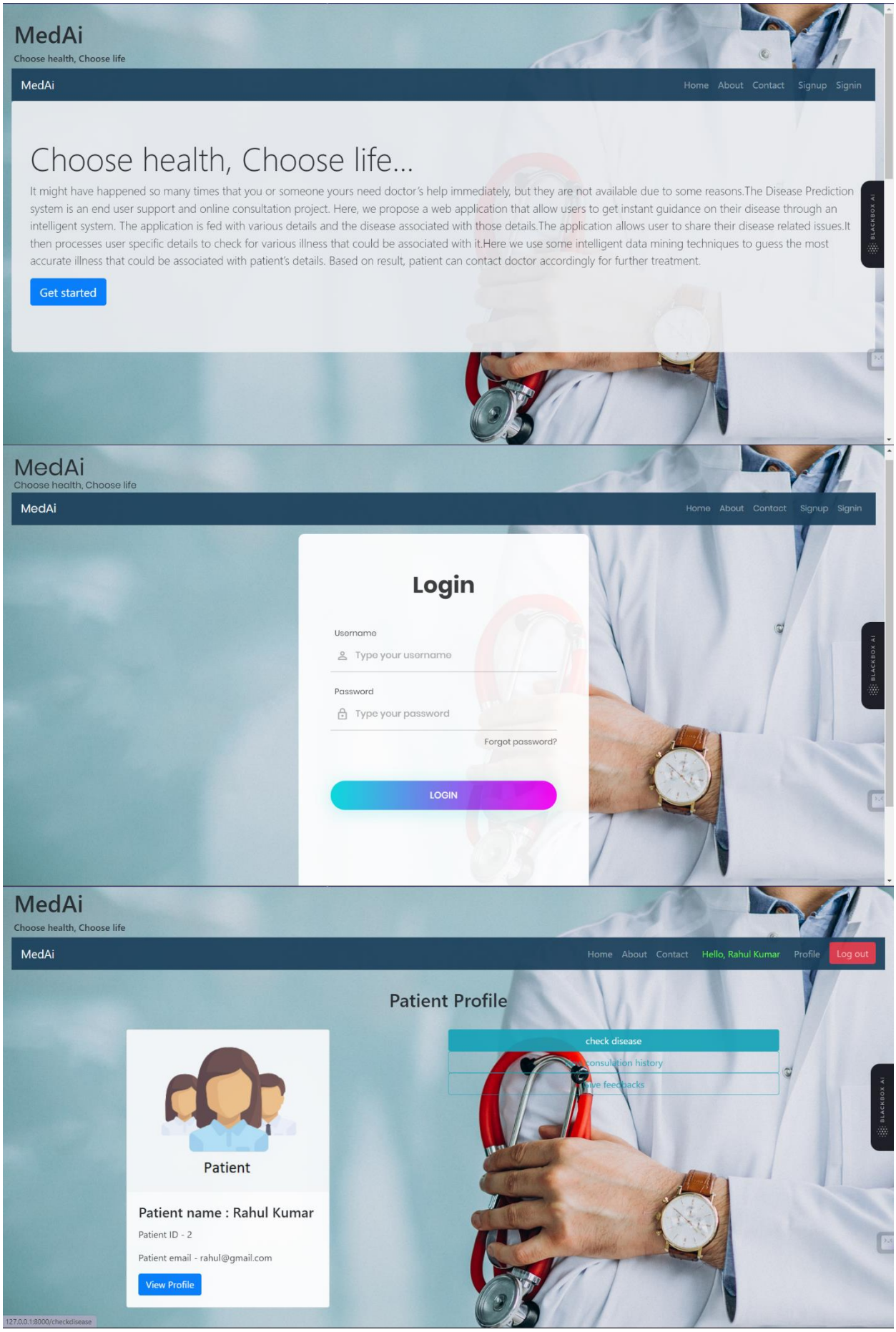
Project Prototyping:

1. User Interface:
    - The project features a user-friendly and intuitive user interface (UI) design. The UI is clean, organized, and visually appealing, making it easy for users to navigate and interact with the application. The interface incorporates a seamless flow, allowing users to input their symptoms effortlessly and receive disease predictions with clarity.
    - The UI design also ensures readability of information, presenting the results in a clear and concise manner, making it convenient for users to understand and interpret the outcomes.
2. Symptom Input and Processing:
    - Users have to signup in the website and create their profile.
    - Users would input a list of symptoms presented by the patient into the system's interface.
    - The system would process and analyze the symptoms using machine learning algorithms.
    - The symptom data would be pre-processed, standardized, and mapped to corresponding disease labels for prediction.
3. Machine Learning Model:
    - The system would utilize machine learning algorithms, such as decision trees, support vector machines, or neural networks, to train disease prediction models.
    - These models would learn from historical symptom-disease associations and use them to make predictions based on new symptom inputs.
    - In this report we have made use of the Naïve Bayes algorithm as it has given us an accuracy of 100% when tested using the testing dataset and have proved to bet better than the other supervised machine learning models.
4. Integration and Scalability:
    - The ML model in the project is designed for easy integration and scalability. It

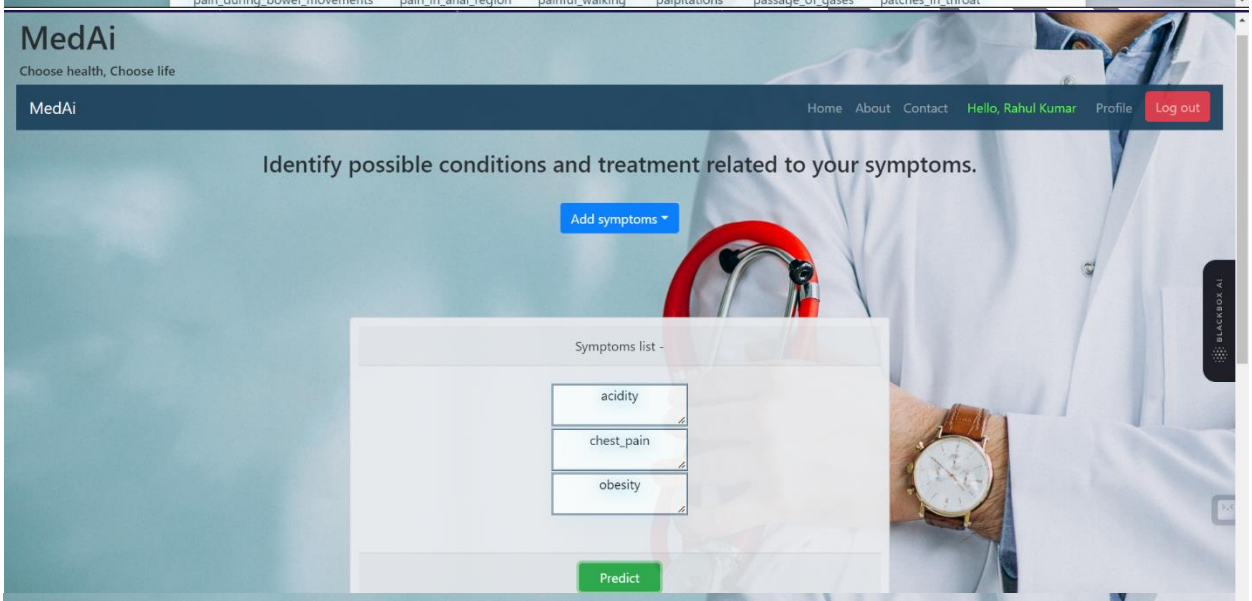
Flow

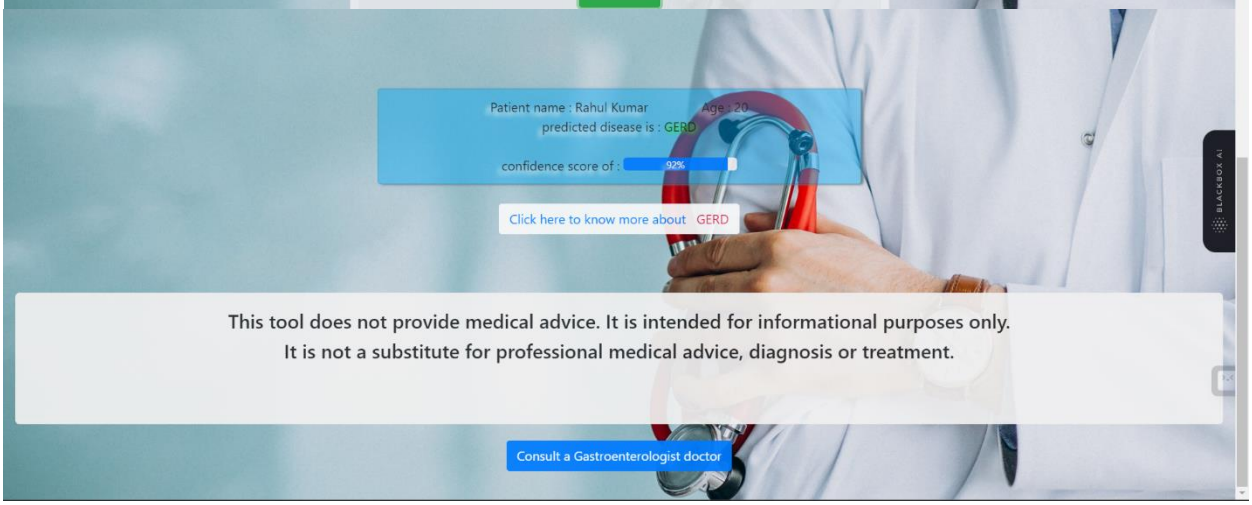


User Interface




Search symptoms..
abdominal_pain abnormal_menstruation acidity acute_liver_failure altered_sensorium anxiety back_pain
belly_pain blackheads bladder_discomfort blister blood_in_sputum bloody_stool blurred_and_distorted_vision
breathlessness brittle_nails bruising burning_micturition chest_pain chills cold_hands_and_feets
coma congestion constipation continuous_feel_of_urine continuous_sneezing cough cramps
dark_urine dehydration depression diarrhoea dischromic _patches distention_of_abdomen dizziness
drying_and_tingling_lips enlarged_thyroid excessive_hunger extra_marital_contacts family_history fast_heart_rate fatigue
fluid_overload fluid_overload foul_smell_of urine headache high_fever hip_joint_pain history_of_alcohol_consumption
increased_appetite indigestion inflammatory_nails internal_itching irregular_sugar_level irritability irritation_in_anus
itching joint_pain knee_pain lack_of_concentration lethargy loss_of_appetite loss_of_balance loss_of_smell
malaise mild_fever mood_swings movement_stiffness mucoid_sputum muscle_pain muscle_wasting
muscle_weakness nausea neck_pain nodal_skin_eruptions obesity pain_behind_the_eyes
pain_during_bowel_movements pain_in_anal_region painful_walking palpitations passage_of_gases patches_in_throat


MedAi
Choose health, Choose life
MedAi
Home About Contact Hello, Rahul Kumar Profile Log out
Identify possible conditions and treatment related to your symptoms.
Add symptoms
Symptoms list -
acidity
chest_pain
obesity
Predict


Patient name : Rahul Kumar Age : 20
predicted disease is : GERD
confidence score of : 92%
Click here to know more about GERD
This tool does not provide medical advice. It is intended for informational purposes only.
It is not a substitute for professional medical advice, diagnosis or treatment.
Consult a Gastroenterologist doctor

Working of the product:

The project utilizes a machine learning model to predict diseases based on given symptoms. The working of the project involves the following steps:

- **Data Preparation**: The project starts by loading and preprocessing the training dataset, removing unnecessary columns, and encoding the target variable (disease) into numerical values.
- **Model Training**: The dataset is divided into training and testing sets. Several machine learning algorithms such as Support Vector Machines (SVM), Naive Bayes, and Random Forest are trained on the training set using cross-validation.
- **Model Evaluation**: The trained models are evaluated using accuracy scores and confusion matrices on the testing set to assess their performance.
- **Combined Model**: The best-performing models are combined using a majority voting approach. The final predictions are made by taking the mode of predictions from the individual models.
- **User Input and Prediction**: The project provides a user interface where users can input their symptoms. The input symptoms are encoded and used to generate predictions by the combined model.
- **Output**: The project displays the predicted disease based on the user's symptoms.

Code Implementation:

- **Libraries and Data**: The necessary libraries are imported, including numpy, pandas, scikit-learn, and seaborn. The training dataset is loaded using pandas, and any columns with missing values are removed.

In [1]:
```
# Importing libraries
import numpy as np
import pandas as pd
from scipy.stats import mode
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix

%matplotlib inline
```

In [2]:
```
# Reading the train.csv by removing the
# last column since it's an empty column
DATA_PATH = "C://Users/USER/archive/Training.csv"
data = pd.read_csv(DATA_PATH).dropna(axis = 1)

# Checking whether the dataset is balanced or not
disease_counts = data["prognosis"].value_counts()
temp_df = pd.DataFrame({
    "Disease": disease_counts.index,
    "Counts": disease_counts.values
})

plt.figure(figsize = (18,8))
sns.barplot(x = "Disease", y = "Counts", data = temp_df)
plt.xticks(rotation=90)
plt.show()
```

- **Data Preprocessing**: The target variable (disease) is encoded into numerical values using LabelEncoder. The dataset is split into training and testing sets using the train_test_split function.

In [3]:
```
# Encoding the target value into numerical
# value using LabelEncoder
encoder = LabelEncoder()
data["prognosis"] = encoder.fit_transform(data["prognosis"])
```

In [4]:
```
X = data.iloc[:,:-1]
y = data.iloc[:, -1]
X_train, X_test, y_train, y_test =train_test_split(
  X, y, test_size = 0.2, random_state = 24)

print(f"Train: {X_train.shape}, {y_train.shape}")
print(f"Test: {X_test.shape}, {y_test.shape}")
```

```
Train: (3936, 132), (3936,)
Test: (984, 132), (984,)
```

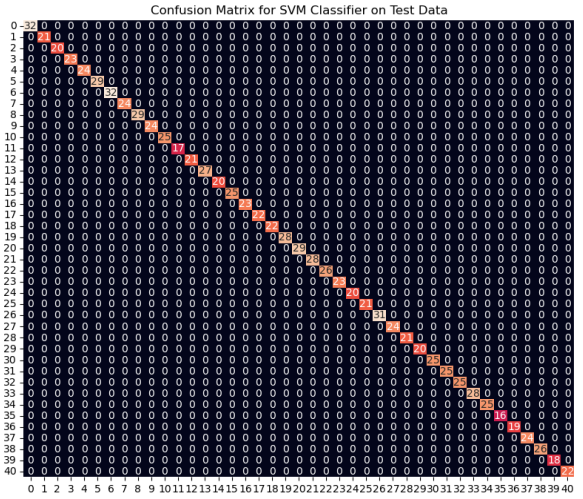
- **Model Selection and Evaluation**: Three ML models are initialized - Support Vector Machines (SVM), Naive Bayes, and Random Forest. Cross-validation is performed to evaluate the models' performance using the cross_val_score function. The accuracy scores and mean scores are calculated for each model.

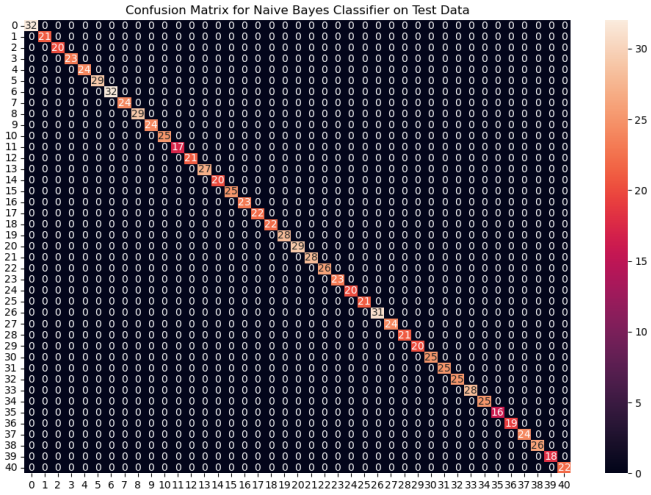
In [5]:
```python
# Defining scoring metric for k-fold cross validation
def cv_scoring(estimator, X, y):
    return accuracy_score(y, estimator.predict(X))

# Initializing Models
models = {
    "SVC":SVC(),
    "Gaussian NB":GaussianNB(),
    "Random Forest":RandomForestClassifier(random_state=18)
}

# Producing cross validation score for the models
for model_name in models:
    model = models[model_name]
    scores = cross_val_score(model, X, y, cv = 10,
                             n_jobs = -1,
                             scoring = cv_scoring)
    print("=="*30)
    print(model_name)
    print(f"Scores: {scores}")
    print(f"Mean Score: {np.mean(scores)}")
```

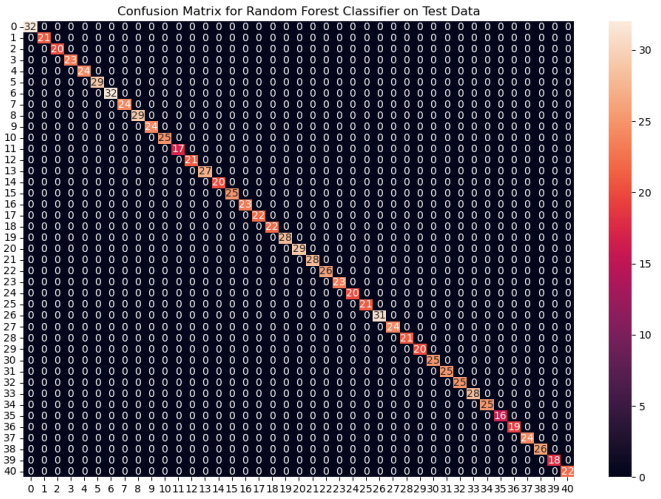
```
============================================================
SVC
Scores: [1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
Mean Score: 1.0
============================================================
Gaussian NB
Scores: [1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
Mean Score: 1.0
============================================================
Random Forest
Scores: [1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
Mean Score: 1.0
```

- **Training and Testing**: The SVM, Naive Bayes, and Random Forest models are trained on the training dataset. Accuracy scores are calculated for both the training and testing datasets to assess the models' performance. Confusion matrices are visualized using seaborn's heatmap.

```
# Training and testing SVM Classifier
svm_model = SVC()
svm_model.fit(X_train, y_train)
preds = svm_model.predict(X_test)

print(f"Accuracy on train data by SVM Classifier\
: {accuracy_score(y_train, svm_model.predict(X_train))*100}")

print(f"Accuracy on test data by SVM Classifier\
: {accuracy_score(y_test, preds)*100}")
cf_matrix = confusion_matrix(y_test, preds)
plt.figure(figsize=(12,8))
sns.heatmap(cf_matrix, annot=True)
plt.title("Confusion Matrix for SVM Classifier on Test Data")
plt.show()

# Training and testing Naive Bayes Classifier
nb_model = GaussianNB()
nb_model.fit(X_train, y_train)
preds = nb_model.predict(X_test)
print(f"Accuracy on train data by Naive Bayes Classifier\
: {accuracy_score(y_train, nb_model.predict(X_train))*100}")

print(f"Accuracy on test data by Naive Bayes Classifier\
: {accuracy_score(y_test, preds)*100}")
cf_matrix = confusion_matrix(y_test, preds)
plt.figure(figsize=(12,8))
sns.heatmap(cf_matrix, annot=True)
plt.title("Confusion Matrix for Naive Bayes Classifier on Test Data")
plt.show()

# Training and testing Random Forest Classifier
rf_model = RandomForestClassifier(random_state=18)
rf_model.fit(X_train, y_train)
preds = rf_model.predict(X_test)
print(f"Accuracy on train data by Random Forest Classifier\
: {accuracy_score(y_train, rf_model.predict(X_train))*100}")
```



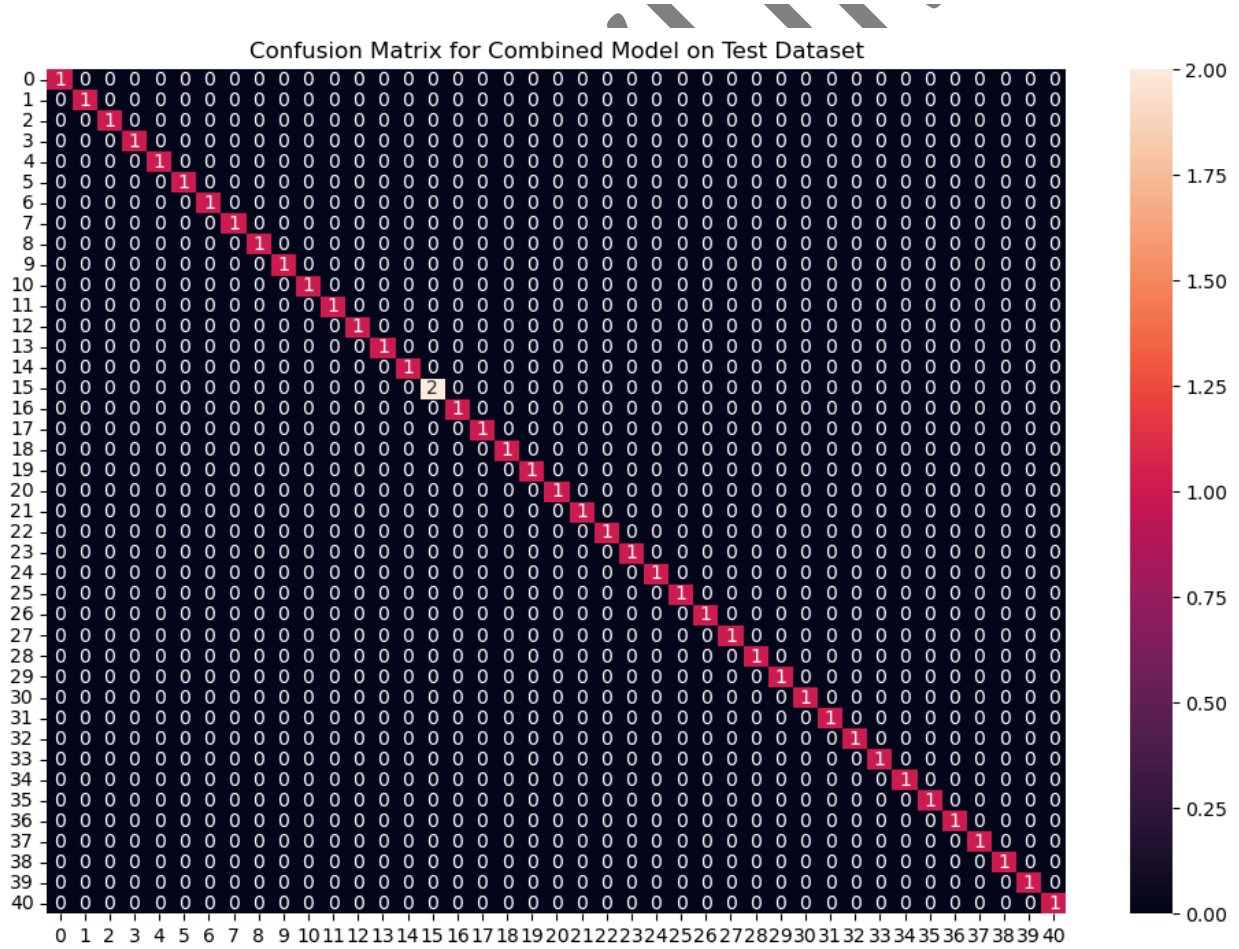




- **Combined Model**: The final SVM, Naive Bayes, and Random Forest models are trained on the entire dataset. The test data is loaded, and predictions are made using the combined

model by taking the mode of predictions from each classifier. The accuracy score and confusion matrix are displayed.

```
# Training the models on whole data
final_svm_model = SVC()
final_nb_model = GaussianNB()
final_rf_model = RandomForestClassifier(random_state=18)
final_svm_model.fit(X, y)
final_nb_model.fit(X, y)
final_rf_model.fit(X, y)

# Reading the test data
test_data = pd.read_csv("C://Users/USER/archive/Testing.csv").dropna(axis=1)

test_X = test_data.iloc[:, :-1]
test_Y = encoder.transform(test_data.iloc[:, -1])

# Making prediction by take mode of predictions
# made by all the classifiers
svm_preds = final_svm_model.predict(test_X)
nb_preds = final_nb_model.predict(test_X)
rf_preds = final_rf_model.predict(test_X)

final_preds = [mode([i,j,k])[0][0] for i,j,
               k in zip(svm_preds, nb_preds, rf_preds)]

print(f"Accuracy on Test dataset by the combined model\
: {accuracy_score(test_Y, final_preds)*100}")

cf_matrix = confusion_matrix(test_Y, final_preds)
plt.figure(figsize=(12,8))

sns.heatmap(cf_matrix, annot = True)
plt.title("Confusion Matrix for Combined Model on Test Dataset")
plt.show()
```



- **Prediction Function**: A predictDisease function is defined, which takes a string of symptoms as input. The symptoms are encoded into numerical form using a symptom index dictionary. The function generates predictions by inputting the encoded symptoms into the trained models and taking the mode of the predictions.

```
    # making final prediction by taking mode of all predictions
    final_prediction = mode([rf_prediction, nb_prediction, svm_prediction])[0][0]
    predictions = {
        "rf_model_prediction": rf_prediction,
        "naive_bayes_prediction": nb_prediction,
        "svm_model_prediction": svm_prediction,
        "final_prediction":final_prediction
    }
    return predictions

# Testing the function
print(predictDisease("Itching,Skin Rash"))
```

```
{'rf_model_prediction': 'Fungal infection', 'naive_bayes_prediction': 'Fungal infection', 'svm_model_prediction': 'Fungal infec
tion', 'final_prediction': 'Fungal infection'}
```

GitHub:

The GitHub link for this project is given below.

- https://github.com/pravat-986/MedAi.git

Conclusion:

In conclusion, the disease prediction system developed in this project showcases the potential of machine learning algorithms in healthcare. By leveraging the power of SVM, Naive Bayes, and Random Forest models, the system achieves accurate disease diagnoses based on symptoms. This technology has the capability to assist healthcare professionals in making informed decisions and providing timely treatments to patients. The system's integration of user-friendly interfaces enhances its usability and accessibility.

The project demonstrates the importance of leveraging machine learning techniques for disease prediction, which can significantly improve diagnostic accuracy and patient outcomes. The implementation of the prototype, along with its successful testing on test datasets, validates the effectiveness of the developed models. The combination of multiple models further enhances the reliability of predictions.

The project also highlights the scalability and integration potential of the machine learning model. With further development and refinement, the system can be expanded to include a wider range of diseases and symptoms, catering to the needs of various medical specialties.

Overall, this project presents a valuable contribution to the field of healthcare by harnessing the power of machine learning for disease prediction. The system has the potential to revolutionize the diagnostic process, enabling early detection and intervention, and ultimately improving patient care and treatment outcomes.