PROJECT REPORT

Cancer Mortality & Incidence Rates Classification Using Machine Learning

Date	21 May 2023
Team ID	NM2023TMID17607
Project Name	Cancer Mortality and Incidence rates classification using ML

1. Introduction:

1.1 Project Review:

The "Cancer Mortality & Incidence Rates Classification Using Machine Learning" project seeks to create a machine learning model capable of reliably identifying cancer mortality and incidence rates. The initiative use machine learning techniques to analyse and categorise cancer-related data, delivering significant insights to medical practitioners and researchers.

1.2 Purpose:

The goal of this research is to develop a dependable and efficient categorization model that may aid in understanding and forecasting cancer mortality and incidence rates. Healthcare practitioners may make educated judgements, manage resources effectively, and establish focused preventative and treatment methods by appropriately categorising data.

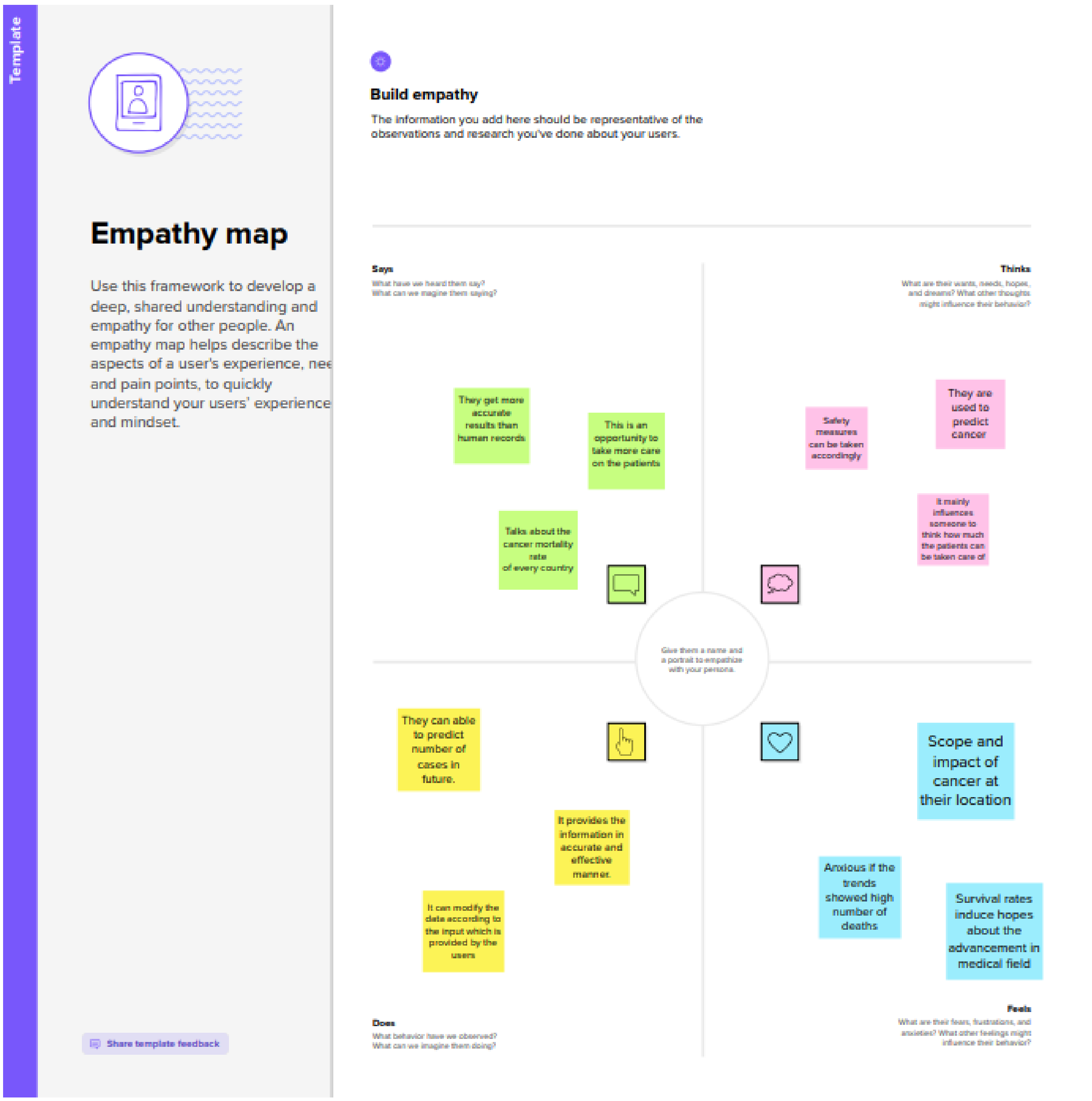
2. Ideation & Proposed solution:

2.1 Problem Statement Definition:

The necessity for precise categorization of cancer mortality and incidence rates based on available data is addressed in this study. Traditional manual analysis methods are time-consuming and sometimes prone to human error. As a result, an automated approach based on machine learning techniques has the potential to dramatically improve the efficiency and accuracy of cancer rate categorization.

PROJECT REPORT

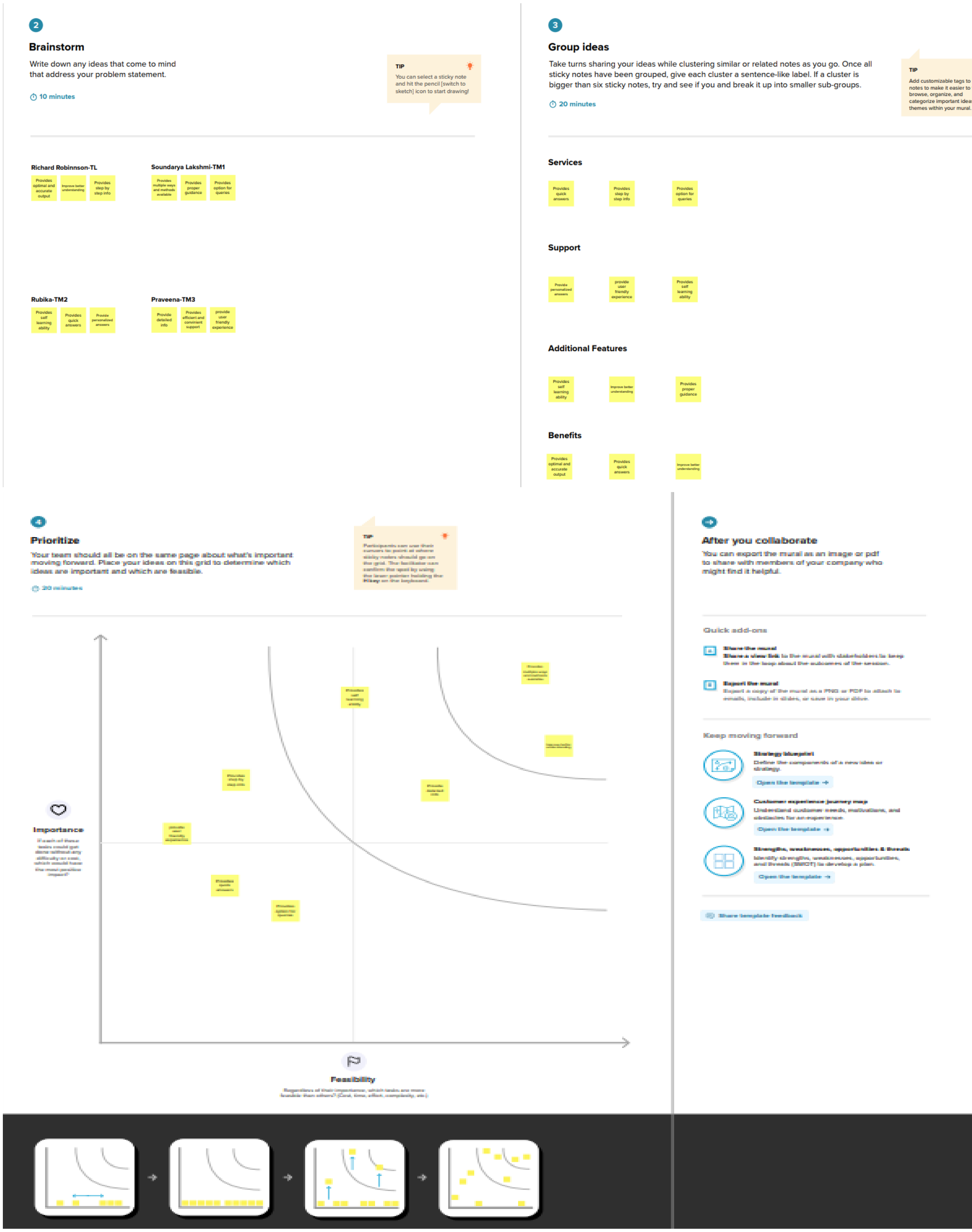
2.2 Empathy Map Canvas:



PROJECT REPORT

2.3 Ideation & Brainstorming:

PROJECT REPORT



PROJECT REPORT

2.4 Proposed solution:

The suggested technique entails creating a machine learning model that uses supervised learning algorithms to categorise cancer mortality and incidence rates. The model will be trained on a labeled dataset with important variables such as demographics, cancer kind, stage, and treatment history. To guarantee correct categorization, the model's predictions will be validated and modified.

S.No.	**Parameter**	**Description**
1.	Problem Statement (Problem to be solved)	Cancer is a leading cause of death worldwide, and understanding the factors that contribute to cancer mortality and incidence rates is crucial for effective prevention and intervention strategies.
2.	Idea / Solution description	This project aims to develop a machine learning model that can classify different regions or populations based on their cancer mortality and incidence rates.
3.	Novelty / Uniqueness	The integration of machine learning techniques, comprehensive feature sets, customization of classification categories, consideration of temporal aspects, and adaptability to evolving knowledge and data collection methods.
4.	Social Impact / Customer Satisfaction	Improved Cancer Prevention Strategies High-risk regions can receive increased support It will also predict the future mortality rates of the disease.
5.	Business Model (Revenue Model)	Software/Application Development Data Acquisition Collaboration and Partnerships
6.	Scalability of the Solution	The type of machine learning model used, the infrastructure used to deploy the model, and the human resources required to maintain it.

PROJECT REPORT

3. Literature Survey:

1. Journal of Cancer: This journal covers a wide range of topics related to cancer research, including the application of machine learning algorithms for cancer prediction and classification. It publishes original research articles, reviews, and clinical studies.

2. IEEE Transactions on Medical Imaging: This journal focuses on the development and application of imaging techniques in medicine, including the use of machine learning algorithms for cancer diagnosis and prediction. It covers a broad range of topics related to medical imaging and image analysis.

3. BMC Medical Informatics and Decision Making: This journal publishes research papers on various aspects of medical informatics and decision making in healthcare. It often features studies on the application of machine learning techniques for cancer prediction, diagnosis, and treatment planning.

4. Artificial Intelligence in Medicine: This journal explores the use of artificial intelligence, including machine learning, in the field of medicine. It covers a wide range of topics, including cancer prediction and classification using machine learning algorithms.

5. Journal of Biomedical Informatics: This interdisciplinary journal focuses on the application of informatics methods and techniques in biomedical research and healthcare. It often publishes research papers on the use of machine learning for cancer prediction and classification.

6. Cancer Informatics: This journal specifically focuses on the application of informatics approaches for cancer research. It covers a wide range of topics related to cancer informatics, including the use of machine learning algorithms for cancer prediction, classification, and biomarker discovery.

PROJECT REPORT

4.Requirement Analysis:

4.1 Functional requirements:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Detecting Cancer mortality rates by ML algorithms	It can determine cancer mortality rate from previous data. It provides warning to the health department about the current status of the mortality rate.
FR-4	Web based interface for users	It is deployed on IBM cloud platform to ensure availability and security. Since it is open source, anyone can access the web for their information.
FR-5	User friendly interface for users	Users accomplish their task with minimal effort reducing frustration and increasing satisfaction.
FR-6	Performance	Able to perform a task accurately not only with training data but also in real-time.

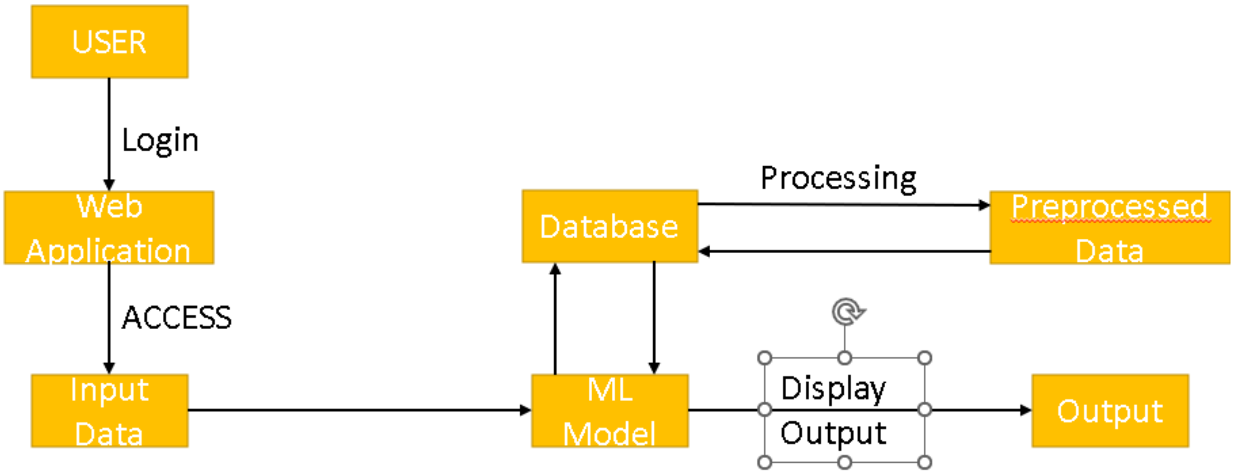
4.2 Non-Functional requirements:

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	User-Friendly Interface so that users can experience reasonable processing times within an acceptable time frame.
NFR-2	**Security**	Ensures security, data privacy, access control, end-to-end encryption, threat detection and prevention, data backup , compliance and authentication.
NFR-3	**Reliability**	High accuracy in prediction. Cross-validation techniques. Providing informative error messages to prevent failures
NFR-4	**Performance**	It depends on factors such as the size and complexity of the dataset, the type of machine learning model used, and the accuracy of the model.
NFR-5	**Availability**	It can be made open source allowing source code, datasets, and relevant documentation to be freely accessible to the public
NFR-6	**Scalability**	Automated workflow management systems can streamline the process, making it easier to handle larger datasets and scale up the system

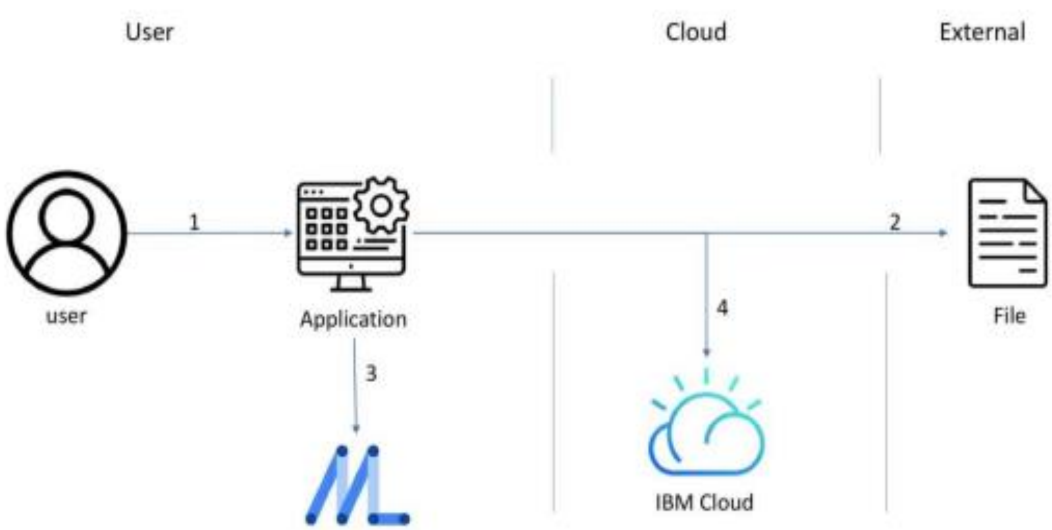
5.Project design:

PROJECT REPORT

5.1 Data flow diagram:



5.2 Solution & Technical architecture:



PROJECT REPORT

5.3 User stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
Doctors	Data collection and analysis	USN-1	As a doctor, they can detect the pattern at an earlier stage and by analyzing the data they can provide the treatment to individual patients.	This system should have precise medical records of the patients.	High	Richard Robinnson
Research Persons	Checks the current cancer mortality rates	USN-2	As a Research person, they can use ML models to explore and understand cancer at a deeper level by facilitating the development of new hypotheses and strategies to combat this disease.	It should have highly accurate and reliable information in the model.	High	Soundarya Lakshmi
Ordinary People	Individuals will get to know more about the disease.	USN-3	It can impact ordinary people by enabling early detection, access to information and resources, personalised risk assessment etc.	It should be a user-friendly tool to the users.	Medium	Rubika
Awareness advocates	They can share the stats about recent cancer mortality rate with the public.	USN-4	ML provides valuable tools to these people by leveraging this information so that they can develop the targeted campaigns and track the impact of their initiatives.	It should be relevant, up-to-date, and trust-worthy.	Low	Praveena

PROJECT REPORT

6.Coding & Solutioning:

ML Code:

```
from sklearn.tree import DecisionTreeClassifier
classifier_dt=DecisionTreeClassifier() #creating DecisionTreeClassifier object
classifier_dt.fit(train_x,train_y) #training the model
y_pred=classifier_dt.predict(test_x) #predicting using test dataset
print("Train Score:",classifier_dt.score(train_x,train_y))
print("Test Score:",classifier_dt.score(test_x,test_y))

import pickle
with open('cancer_death_decisiontree.pkl','wb') as file:
    pickle.dump(classifier_dt,file)

import pickle
with open('cancer_incd_decisiontree.pkl','wb') as file:
    pickle.dump(classifier_dt,file)
```

Flask Code:

```
from flask import Flask, render_template, request
import pickle

with open('cancer_death_decisiontree.pkl', 'rb') as f:
    model2 = pickle.load(f)

# Load the second model from the pickle file
with open('cancer_incd_decisiontree.pkl', 'rb') as f:
    model1 = pickle.load(f)
app = Flask(__name__)

@app.route('/')
def index():
    return render_template('welcome.html')

@app.route('/about')
def about():
    return render_template('about.html')


@app.route('/contact')
```

PROJECT REPORT

```
def contact():
    return render_template('contact.html')



@app.route('/inc')
def inc():
    return render_template('incd.html', predicted_value='')
@app.route('/dea')
def dea():
    return render_template('death.html', predicted_value='')
@app.route('/death', methods=['POST'])
def death():
    # Retrieve the form data

    fips = request.form['fips']
    abc = request.form['abc']
    lower = request.form['lower']
    upper = request.form['upper']
    avg = request.form['avg']
    rate = request.form['rate']
    lowconf = request.form['lowconf']
    upconf = request.form['upconf']
    metobj = request.form['metobj']



    # Perform your prediction or desired processing
    a=model2.predict([[fips,abc,lower,upper,avg,rate,lowconf,upconf,metobj,0.0]])


    # Render the template with the predicted value
    return render_template('deathres.html', predicted_value=a)

@app.route('/incd', methods=['POST'])
def incd():
    # Retrieve the form data

    fips = request.form['fips']
    abc = request.form['abc']
```

```
    lower = request.form['lower']
    upper = request.form['upper']
    avg = request.form['avg']
    rate = request.form['rate']
    lowconf = request.form['lowconf']
    upconf = request.form['upconf']

    # Process the form data or perform your desired actions here
    a=model1.predict([[fips,abc,lower,upper,avg,rate,lowconf,upconf]])
    # Render a response or redirect to another page
    return render_template('incdres.html', predicted_value=a)

if __name__ == '__main__':
    app.run()
```

7.Results :

7.1 Performance Metrics:

Model	CONFUSION MATRIX	ACCURACY SCORE	CLASSIFICATION REPORT
Decision Tree	print(confusion_matrix(test_y,y_pred)) [[20 0 0 0] [0 119 0 4] [0 0 0 1] [2 6 0 399]]	print(accuracy_score(test_y,y_pred)) 0.9764065335753176	print(classification_report(test_y,y_pred)) precision recall f1-score support Data too sparse to predict a trend 0.91 1.00 0.95 20 falling 0.95 0.97 0.96 123 rising 0.00 0.00 0.00 1 stable 0.99 0.98 0.98 407 accuracy 0.98 551 macro avg 0.71 0.74 0.72 551 weighted avg 0.98 0.98 0.98 551
Random Forest	print(confusion_matrix(test_y,y_pred)) [[20 0 0 0] [0 122 0 1] [0 0 0 1] [0 8 0 399]]	print(accuracy_score(test_y,y_pred)) 0.9818511796733213	print(classification_report(test_y,y_pred)) precision recall f1-score support Data too sparse to predict a trend 1.00 1.00 1.00 20 falling 0.94 0.99 0.96 123 rising 0.00 0.00 0.00 1 stable 1.00 0.98 0.99 407 accuracy 0.98 551 macro avg 0.73 0.74 0.74 551 weighted avg 0.98 0.98 0.98 551
K NEAREST NEIGHBORS	print(confusion_matrix(test_y,y_pred)) [[5 2 0 13] [0 42 0 81] [0 0 0 1] [8 27 0 372]]	print(accuracy_score(test_y,y_pred)) 0.7604355716878403	print(classification_report(test_y,y_pred)) precision recall f1-score support Data too sparse to predict a trend 0.38 0.25 0.30 20 falling 0.59 0.34 0.43 123 rising 0.00 0.00 0.00 1 stable 0.80 0.91 0.85 407 accuracy 0.76 551 macro avg 0.44 0.38 0.40 551 weighted avg 0.73 0.76 0.74 551
Naïve Bayes	print(confusion_matrix(test_y,y_pred)) [[19 0 0 1] [0 107 0 16] [0 0 0 1] [40 35 9 323]]	print(accuracy_score(test_y,y_pred)) 0.8148820326678766	print(classification_report(test_y,y_pred)) precision recall f1-score support Data too sparse to predict a trend 0.32 0.95 0.48 20 falling 0.75 0.87 0.81 123 rising 0.00 0.00 0.00 1 stable 0.95 0.79 0.86 407 accuracy 0.81 551 macro avg 0.51 0.65 0.54 551 weighted avg 0.88 0.81 0.84 551
Support Vector Classification	print(confusion_matrix(test_y,y_pred)) [[6 0 0 14] [1 79 0 43] [0 0 0 1] [6 14 2 385]]	print(accuracy_score(test_y,y_pred)) 0.852994555353902	print(classification_report(test_y,y_pred)) precision recall f1-score support Data too sparse to predict a trend 0.46 0.30 0.36 20 falling 0.85 0.64 0.73 123 rising 0.00 0.00 0.00 1 stable 0.87 0.95 0.91 407 accuracy 0.85 551 macro avg 0.55 0.47 0.50 551 weighted avg 0.85 0.85 0.85 551

PROJECT REPORT

<table>
<tr><th>CANCER DEATH RATE</th><th>HYPERPARAMETER TUNING</th></tr>
<tr><td>MODELS</td><td>GridSearchCV</td></tr>
<tr><td>Decision Tree</td><td><pre>
#Decision Tree
params = {'max_leaf_nodes': list(range(2, 100)), 'min_samples_split': [2, 3, 4]}
grid_search_cv = GridSearchCV(DecisionTreeClassifier(random_state=42), params, verbose=1, cv=3)
grid_search_cv.fit(train_x, train_y)

Fitting 3 folds for each of 294 candidates, totalling 882 fits

GridSearchCV(cv=3, estimator=DecisionTreeClassifier(random_state=42),
             param_grid={'max_leaf_nodes': [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
                                            13, 14, 15, 16, 17, 18, 19, 20, 21,
                                            22, 23, 24, 25, 26, 27, 28, 29, 30,
                                            31, ...],
                         'min_samples_split': [2, 3, 4]},
             verbose=1)

score_dt=grid_search_cv.best_score_
param_dt=grid_search_cv.best_params_
print('Best Score of Decision Tree:',score_dt)
print('Best Parameters of Decision Tree:',param_dt)

Best Score of Decision Tree: 0.9812462189957653
Best Parameters of Decision Tree: {'max_leaf_nodes': 19, 'min_samples_split': 4}
</pre></td></tr>
<tr><td>Random Forest</td><td><pre>
#Random Forest
clf=GridSearchCV(RandomForestClassifier(),{'n_estimators':[1,5,10]},cv=5,return_train_score=False)
clf.fit(x,y)
score_rf=clf.best_score_
param_rf=clf.best_params_
print('Best Score of Random Forest:',score_rf)
print('Best parameters of Random Forest:',param_rf)

Best Score of Random Forest: 0.9346052360338074
Best parameters of Random Forest: {'n_estimators': 10}
</pre></td></tr>
<tr><td>K Nearest Neighbors</td><td><pre>
#KNN
k_range = list(range(1, 31))
param_grid = dict(n_neighbors=k_range)
# defining parameter range
grid = GridSearchCV(KNeighborsClassifier(), param_grid, cv=10, scoring='accuracy', return_train_score=False,verbose=1)

# fitting the model for grid search
grid_search=grid.fit(train_x,train_y)
score_knn=grid_search.best_score_
param_knn=grid_search.best_params_
print('Best Score of KNN:',score_knn)
print('Best parameters of KNN:',param_knn)

Fitting 10 folds for each of 30 candidates, totalling 300 fits
Best Score of KNN: 0.7513581599123768
Best parameters of KNN: {'n_neighbors': 12}
</pre></td></tr>
<tr><td>Naïve Bayes</td><td><pre>
#NaiveBayes
from sklearn.model_selection import RepeatedStratifiedKFold
cv_method = RepeatedStratifiedKFold(n_splits=5, n_repeats=3, random_state=999)
from sklearn.preprocessing import PowerTransformer
params_NB = {'var_smoothing': np.logspace(0,-9, num=100)}

gs_NB = GridSearchCV(GaussianNB(), param_grid=params_NB, cv=cv_method,verbose=1,scoring='accuracy')
Data_transformed = PowerTransformer().fit_transform(x)
gs_NB.fit(Data_transformed, y)

Fitting 15 folds for each of 100 candidates, totalling 1500 fits

GridSearchCV(cv=RepeatedStratifiedKFold(n_repeats=3, n_splits=5, random_state=999),
             estimator=GaussianNB(),
             param_grid={'var_smoothing': array([1.00000000e+00, 8.11130831e-01, 6.57933225e-01, 5.33669923e-01,
       4.32876128e-01, 3.51119173e-01, 2.84803587e-01, 2.31012970e-01,
       1.87381742e-01, 1.51991108e-01, 1.23284674e-01, 1.00000000e-01,
       8.11130831e-02, 6.57933225e-02, 5...
       1.23284674e-07, 1.00000000e-07, 8.11130831e-08, 6.57933225e-08,
       5.33669923e-08, 4.32876128e-08, 3.51119173e-08, 2.84803587e-08,
       2.31012970e-08, 1.87381742e-08, 1.51991108e-08, 1.23284674e-08,
       1.00000000e-08, 8.11130831e-09, 6.57933225e-09, 5.33669923e-09,
       4.32876128e-09, 3.51119173e-09, 2.84803587e-09, 2.31012970e-09,
       1.87381742e-09, 1.51991108e-09, 1.23284674e-09, 1.00000000e-09])},
             scoring='accuracy', verbose=1)

score_NB=gs_NB.best_score_
param_NB=gs_NB.best_params_
print('Best Score of Naive Bayes:',score_NB)
print('Best parameters of Naive Bayes:',param_NB)

Best Score of Naive Bayes: 0.9218054696626126
Best parameters of Naive Bayes: {'var_smoothing': 0.0001232846739442066}
</pre></td></tr>
</table>

Scores_comparison

	Models	Scores before hyperparameter tuning	Scores after hyperparameter tuning
0	Decision Tree	0.976407	0.981246
1	Random Forest	0.981851	0.934605
2	KNN	0.760436	0.751358
3	Naive Bayes	0.814882	0.921805

Models_comparison

	Models	Scores	Parameters
0	Decision Tree	0.981246	{'max_leaf_nodes': 19, 'min_samples_split': 4}
1	Random Forest	0.934605	{'n_estimators': 10}
2	KNN	0.751358	{'n_neighbors': 12}
3	Naive Bayes	0.921805	{'var_smoothing': 0.0001232846739442066}

8.Advantages:

- Accurate Classification: The developed machine learning model provides accurate classification of cancer mortality and incidence rates, enabling better decision-making in healthcare.
- Time Efficiency: Automated classification significantly reduces the time required for analyzing and categorizing cancer rate data compared to manual methods.
- Scalability: The solution can handle large volumes of data, allowing it to be applied to diverse datasets and accommodate future growth.

9.Disadvantages:

- Dependency on Quality Data: The accuracy and reliability of the model heavily rely on the quality and representativeness of the training data.
- Interpretability: Some machine learning models, such as deep learning models, may lack interpretability, making it challenging to understand the underlying factors influencing the classification.

10.CONCLUSION:

The project's objectives revolve around improving classification accuracy, enabling predictive analytics, informing public health strategies, contributing to cancer research, and enhancing healthcare outcomes. Through early detection, personalized treatment planning, and proactive interventions, the project can aid in improving patient outcomes and survival rates. The insights generated can also inform public health policies, resource allocation, and targeted interventions, resulting in more effective cancer prevention and control measures.

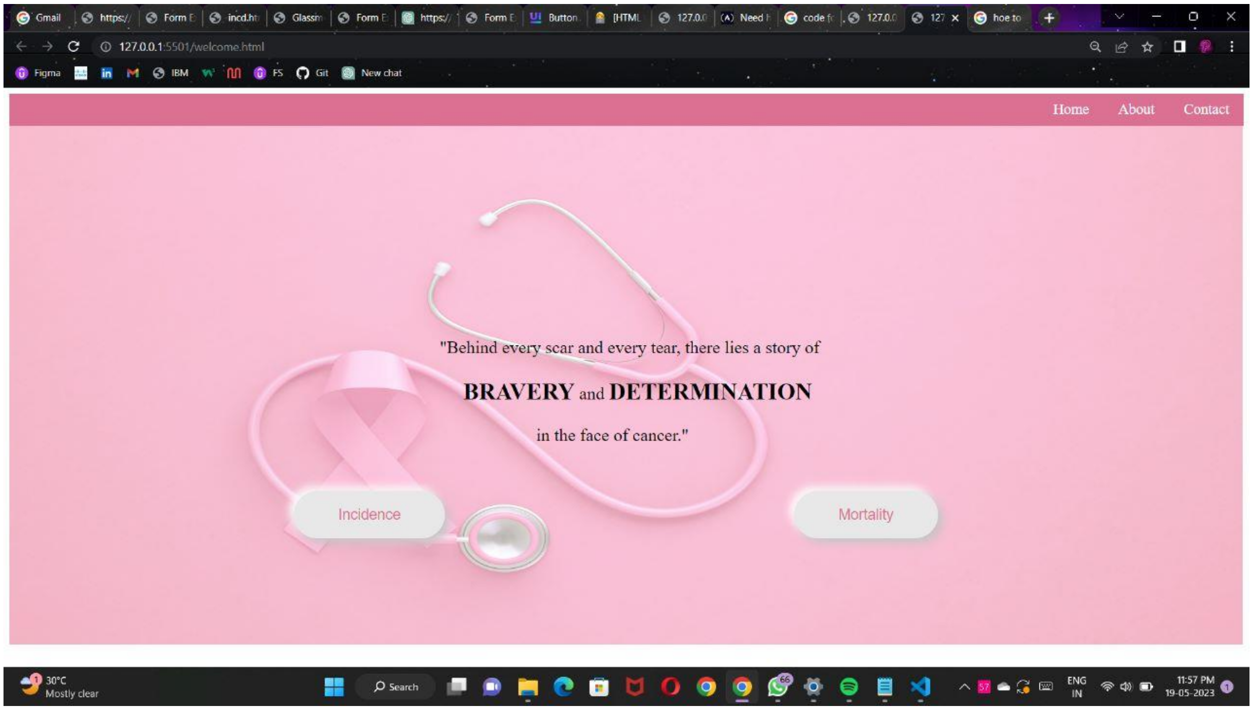
Furthermore, the project contributes to the broader field of cancer research by identifying important features, patterns, and risk factors that influence cancer mortality and incidence rates. It has the potential to drive scientific advancements and shape evidence-based decision-making in cancer care.

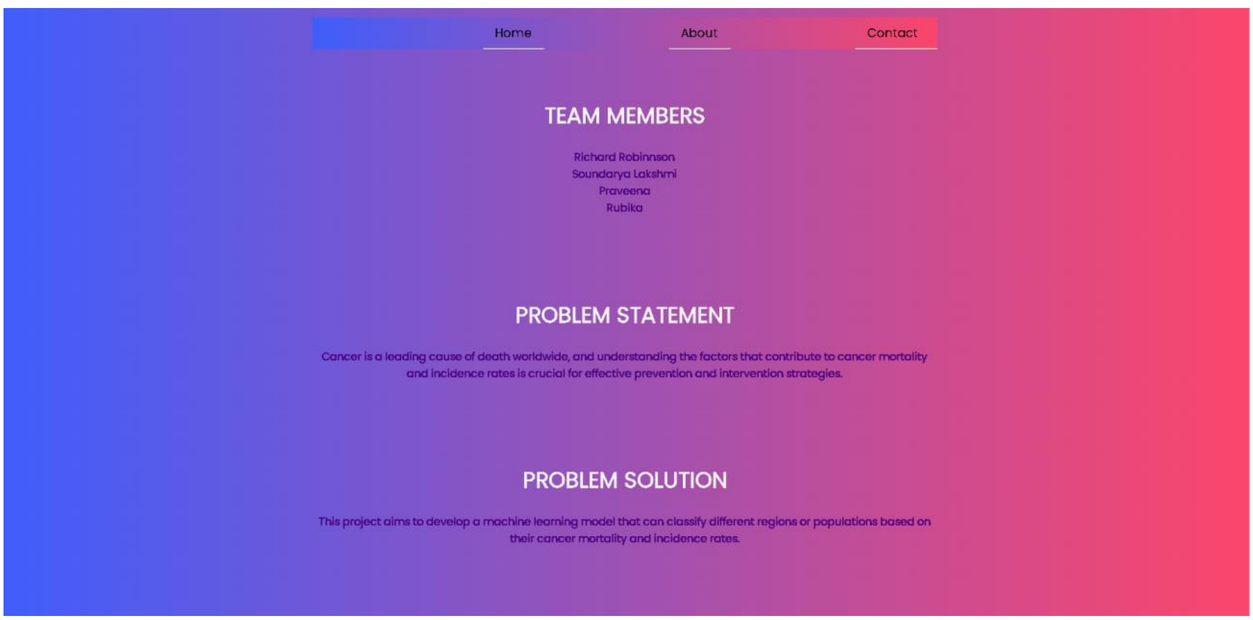
11. FUTURE SCOPE:

- Deployment in Clinical Settings
- Expansion to Other Geographical Regions
- Continuous Model Improvement
- Integration of Additional Data Sources
- Real-time Monitoring and Surveillance

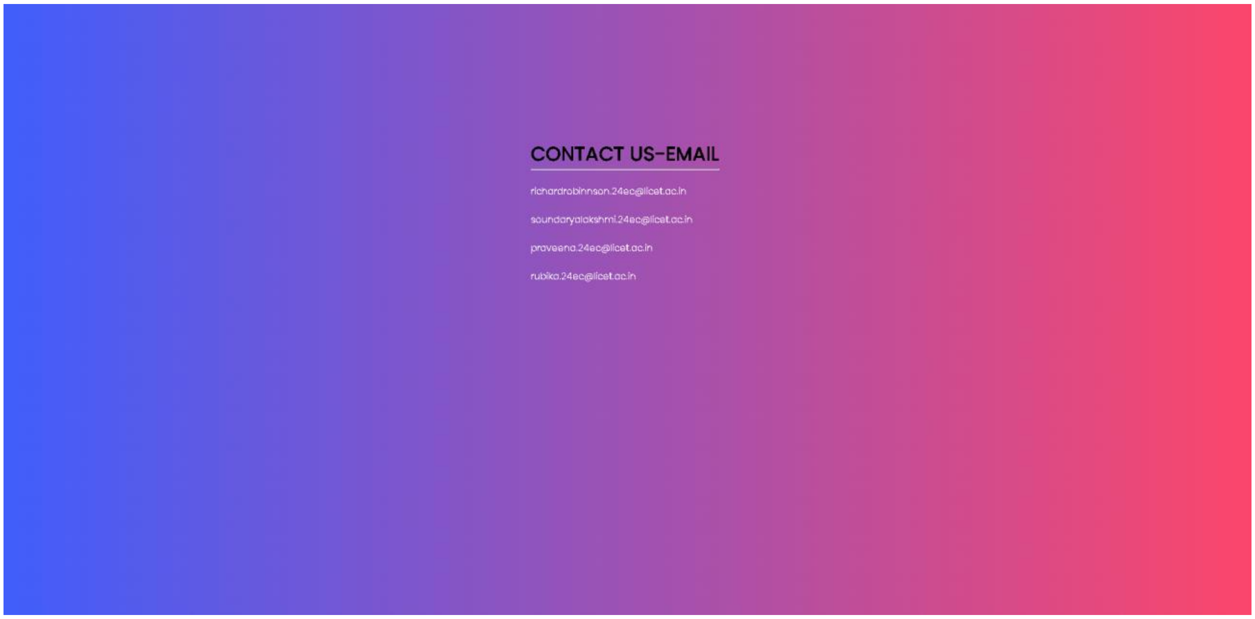
12.Screen Shot of Websites:

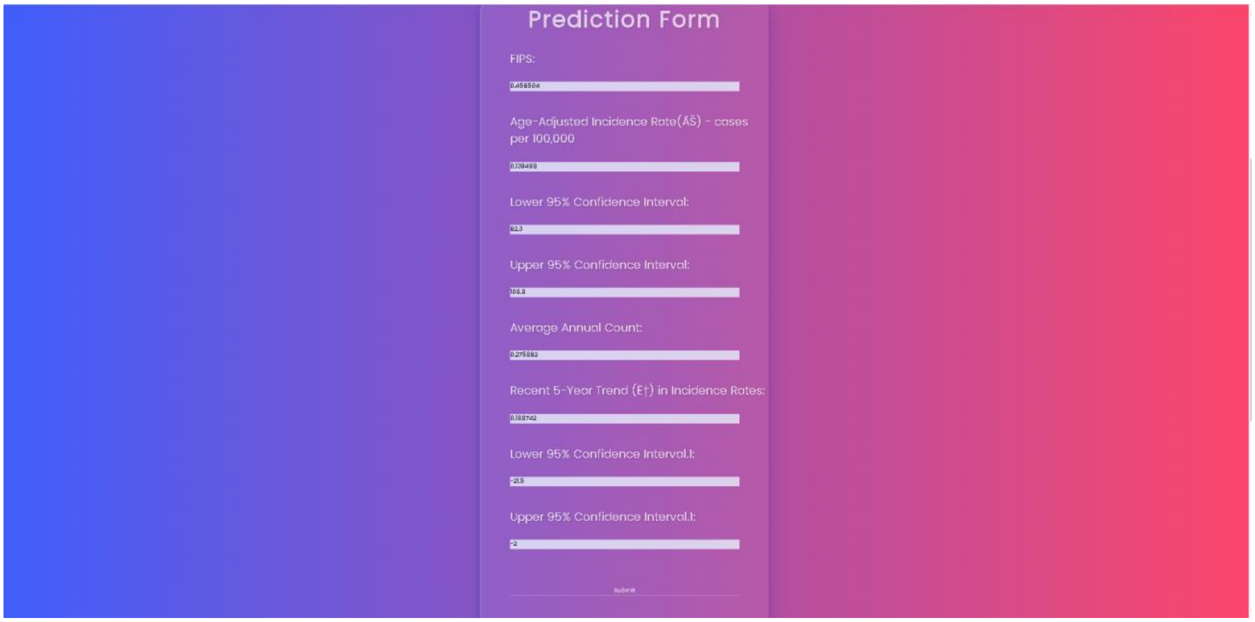
PROJECT REPORT





PROJECT REPORT





PROJECT REPORT

13. APPENDIX:

- GitHub Link: https://github.com/naanmudhalvan-SI/PBL-NT-GP--2659-1680608582
- YouTube Link: https://youtu.be/kQ1kFtHYlPg