Contents

1. Predictive Modelling of Teenage Pregnancy Risk Based on Socio-Economic and Behavioural Factors

1.1 Problem Definition

1.1.1 Problem Statement

This project proposes the development of a predictive machine learning model aimed at assessing the risk of teenage pregnancy based on socio-economic, educational, and behavioural factors. By leveraging survey and administrative data, the model will identify individuals at elevated risk, facilitating proactive interventions, targeted resource allocation, and evidence-based policy design.

1.1.2 Objectives

1. Enable early identification of adolescents at elevated risk for pregnancy, allowing for timely, targeted health education and counselling interventions.
2. Support public health planning and policy formulation through robust, data-driven risk assessment tools.
3. Reduce teenage pregnancy rates by optimizing resource allocation and intervention design for high-risk populations.

1.3 Stakeholders

- **Government Health Departments:** Responsible for reproductive health initiatives, public health campaigns, and funding allocations.
- **Educational Institutions and Policy-Makers:** School boards and administrators interested in improving student well-being, retention rates, and educational outcomes.

1.4 Key Performance Indicator (KPI)

- **Predictive Accuracy:** The percentage of correctly classified high-risk and low-risk cases, serving as an overall measure of the model's effectiveness in identifying individuals in need of intervention.

1.2. Data Collection and Pre-processing

1.2.1 Data Sources

1. **National Health and Demographic Surveys:** These datasets typically include variables such as age, household income, parental education levels, access to health services, sexual behaviour, and contraceptive use.

2. **School Administrative Records:** Data may include attendance rates, academic performance, disciplinary records, participation in school-based sexual education programs, and counselling notes.

1.2.2 Potential Bias in the Data

One key source of bias arises from **underreporting due to social desirability and stigma**. Adolescents may conceal or misreport sexual activity and pregnancy history in surveys. This behaviour can result in data that underrepresents actual incidence rates, thereby skewing model learning and reducing predictive validity.

1.2.3 Pre-processing Steps

Handling Missing Data:

> Apply appropriate imputation strategies to manage incomplete records. Numerical variables can use mean or median imputation, while categorical variables may employ mode imputation or more advanced model-based methods to preserve data integrity.

Encoding Categorical Variables:

> Transform variables such as parental education level, school type, or geographic location into numerical formats suitable for machine learning, using techniques such as one-hot encoding or ordinal encoding where appropriate.

Normalization:

> Scale numerical variables (e.g., age, household income) using standardization or min–max scaling to ensure uniform variable contribution and mitigate issues arising from differing value ranges during model training.

1.3. Model Development

1.3.1 Model Choice and Justification

A **Random Forest Classifier** is selected for this predictive task. Random Forests are ensemble methods that combine multiple decision trees to improve accuracy and robustness. They are well-suited for heterogeneous tabular data with mixed categorical and numerical features. Moreover, they provide interpretable results through feature importance measures, which is vital for understanding and justifying the drivers of risk in a policy context.

1.3.2 Data Splitting Strategy

The dataset will be partitioned as follows to support model development and unbiased evaluation:

- **Training Set (70%)**: Used to train the model by learning underlying patterns in the data.

- **Validation Set (15%)**: Employed during model tuning to evaluate hyper-parameter choices and avoid overfitting.
- **Test Set (15%)**: Held out from all training and validation steps, providing an unbiased final estimate of generalization performance.

1.3.3 Hyper-parameters to Tune

1. **Number of Trees (estimators)**: Controls the size of the ensemble. Increasing the number can improve stability and predictive power but with added computational cost. Tuning helps identify the optimal trade-off between accuracy and efficiency.
2. **Maximum Tree Depth (max_depth)**: Limits the depth of each decision tree, preventing overfitting to noise in the training data. Tuning this parameter is critical for maintaining generalizability to new, unseen data.

1.4. Evaluation and Deployment

1.4.1 Evaluation Metrics

- **Accuracy:** Measures the proportion of correctly classified examples in the dataset. While easy to interpret, it can be misleading if classes are imbalanced (e.g., fewer positive pregnancy cases), making it necessary to consider additional metrics.
- **F1-Score:** The harmonic mean of precision and recall. This metric is particularly valuable in scenarios with class imbalance, as it balances the trade-off between false positives and false negatives. In public health, reducing false negatives (missing high-risk individuals) is critical.

1.4.2 Concept Drift and Monitoring

Concept drift refers to changes over time in the statistical properties of the input variables or their relationship with the target variable. For example, evolving socio-economic conditions, changes in sexual education curricula, or shifts in cultural attitudes can alter the factors influencing teenage pregnancy risk.

To monitor for drift post-deployment:

- Continuously track model performance metrics on new data.
- Conduct periodic evaluations using updated datasets.
- Retrain or update the model as required to maintain predictive accuracy and relevance to current conditions.

1.4.3 Technical Challenge During Deployment

Scalability:
Deploying the predictive model at scale—across diverse regions, schools, or health systems—poses significant challenges. These include ensuring sufficient computational resources to handle large volumes of data in near real-time, maintaining data privacy and security (especially for sensitive health and behavioural information), and integrating the model within existing health information systems or school administration platforms. Addressing these issues requires robust infrastructure planning and potentially cloud-based deployment strategies to ensure broad, secure, and efficient access.

2. Predictive Modelling of Patient Readmission Risk within 30 Days of Discharge

2.1. Problem Scope

2.1.1 Problem Definition

This project proposes the development of a machine learning-based predictive system designed to assess the **risk of hospital patient readmission within 30 days of discharge**. Unplanned readmissions are widely recognized as key indicators of healthcare quality and often stem from factors such as complications, inadequate discharge planning, poor medication management, or insufficient follow-up care. By leveraging clinical, demographic, and administrative data, the model will enable healthcare providers to identify patients at elevated risk and design tailored interventions aimed at reducing avoidable readmissions, improving patient outcomes, and lowering healthcare costs.

2.1.2 Objectives

1. **Enable Early Risk Identification:** Facilitate the identification of patients at high risk of readmission at the time of discharge, supporting proactive care planning.
2. **Support Targeted Interventions:** Inform the development of personalised post-discharge care strategies, such as medication reconciliation, telehealth monitoring, home health visits, or scheduled follow-up appointments.
3. **Improve Quality of Care:** Contribute to hospital quality-improvement initiatives by reducing 30-day readmission rates, lowering associated penalties, and enhancing overall patient safety and satisfaction.

2.1.3 Stakeholders

- **Healthcare Providers:** Physicians, nurses, care coordinators, and discharge planners who will utilise risk predictions to design and implement patient-specific intervention plans.
- **Hospital Administration:** Leadership teams responsible for strategic planning, regulatory compliance, cost management, and quality metrics, including hospital readmission reduction targets set by healthcare authorities and payers.

2.2. Data Strategy

2.2.1 Proposed Data Sources

For our real implementation with diabetic_data.csv, we used:

1. **Electronic Health Records (EHRs):**
   - Diagnoses (ICD codes)
   - Number of inpatient admissions
   - Number of emergency visits
   - Discharge disposition
   - Number of lab procedures and medications
   - Time in hospital
   - Whether diabetes medications were prescribed
2. **Demographic and Utilization Data:**
   - Age bands, gender (where available)
   - Historical hospital usage patterns

Example Code:

```
df = pd.read_csv('/content/diabetic_data.csv')

df.head()
```

2.2.2 Ethical Concerns

1. **Patient Privacy:**
   - EHR data contains highly sensitive information.
   - Requires robust encryption, access controls, and compliance with regulations (e.g., HIPAA, POPIA, GDPR).
2. **Bias and Health Equity:**
   - Models may learn from historically biased data.
   - Risk of unfairly labeling vulnerable populations as "high risk" without addressing systemic disparities.

2.2.3 Preprocessing Pipeline

Our real pipeline in Colab included:

1. Data Cleaning:

```
df = df.replace('?', pd.NA)
missing = df.isnull().mean()
df = df.drop(columns=missing[missing > 0.3].index)
df = df.dropna()
```

- Replaced ambiguous values, dropped sparse columns and incomplete rows.

2. Target Engineering:

```
df['readmitted_30d'] = df['readmitted'].apply(lambda x: 1 if x == '<30' else 0)
```

- Converted multi-class readmission label to binary.

3. Drop IDs:

```
df = df.drop(['encounter_id', 'patient_nbr', 'readmitted'], axis=1)
```

- Avoided data leakage.

4. Encoding:

```
df_encoded = pd.get_dummies(df, drop_first=True)
```

- Handled categorical variables.

5. Scaling:

```
scaler = StandardScaler()
X_scaled = pd.DataFrame(scaler.fit_transform(X), columns=X.columns)
```

- Ensured model convergence.

6. Splitting:

X_train, X_test, y_train, y_test = train_test_split(

X_scaled, y, test_size=0.2, random_state=42, stratify=y

)

- Preserved class balance in training/testing sets.

Feature Importance Example Output:
(from model)

number_inpatient, discharge_disposition_id, number_emergency, diag_1_V58

- Clinically meaningful drivers of readmission risk.

2.3. Model Development

2.3.1 Model Selection and Justification

We selected **Gradient Boosting Classifier**:

from sklearn.ensemble import GradientBoostingClassifier

model = GradientBoostingClassifier(n_estimators=100, max_depth=3, random_state=42)

model.fit(X_train, y_train)

Justification:

- Handles mixed data types well.
- Captures non-linear relationships.
- Provides interpretable feature importance.
- Strong performance on structured tabular data like EHRs.

2.3.2 Confusion Matrix and Precision/Recall (Actual Results)

Code:

from sklearn.metrics import confusion_matrix, classification_report, precision_score, recall_score

```
y_pred = model.predict(X_test)

cm = confusion_matrix(y_test, y_pred)

print(cm)
```

Actual Confusion Matrix:

```
[[17377    21]
 [ 2194    19]]
```

Precision & Recall:

Precision: 0.47

Recall: 0.01

Classification Report Snippet:

	precision	recall	f1-score	support
0	0.89	1.00	0.94	17398
1	0.47	0.01	0.02	2213

Interpretation:

- Model is excellent at identifying *non-readmissions* (high specificity).
- Extremely low *recall* for readmitted patients (only 1% detected).
- This highlights class imbalance and risk of missing true positives—unacceptable in clinical deployment without further work.

2.4. Deployment

2.4.1 Integration Steps

1. **API Deployment:**
    - Expose trained model via REST API (Flask/FastAPI).
    - Accept patient discharge data as JSON input.
    - Return risk score/prediction.
2. **Integration with Hospital EHR System:**
    - Connect API to hospital's EMR system (e.g., Epic, Cerner).
    - Automatically trigger predictions at discharge.
    - Display risk scores in clinician dashboard.

3. **Clinician Workflow:**
    - Train staff on interpreting predictions.
    - Embed risk flags in discharge planning workflow.
    - Provide options for recommended interventions.

2.4.2 Compliance with Healthcare Regulations

- Encrypt data in transit (TLS/HTTPS) and at rest.
- Log all access for auditability.
- Role-based access control to restrict sensitive predictions.
- Maintain HIPAA compliance by ensuring PHI is used only for authorized clinical purposes.
- Conduct regular security assessments.

2.5. Optimization

2.5.1 Method to Address Overfitting

Proposed Method:
Use **Cross-Validation with Hyperparameter Tuning**:

```
from sklearn.model_selection import GridSearchCV


param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [3, 5, 7],
    'learning_rate': [0.01, 0.1, 0.2]
}


grid_search = GridSearchCV(
    estimator=GradientBoostingClassifier(random_state=42),
    param_grid=param_grid,
    cv=5,
    scoring='recall'
```

)

grid_search.fit(X_train, y_train)

Explanation:

- GridSearchCV systematically tests hyperparameters to find the best generalizable model.
- Cross-validation ensures model performance is robust to new data.
- Optimizing for **recall** helps mitigate the extreme imbalance and improves sensitivity to actual readmissions.Part 3: Critical Thinking

3.1 Ethics & Bias

3.1.1 How Might Biased Training Data Affect Patient Outcomes in the Case Study?

Predictive models trained on Electronic Health Records (EHR) data risk learning and replicating **systemic biases** present in historical care patterns. These biases may arise from disparities in access, quality of care, or clinician decision-making. For example:

- **Socio-economic Bias:** Patients from low-income backgrounds may show higher historical readmission rates due to barriers to outpatient care, limited social support, or transportation issues—not because of clinical factors alone.
- **Racial and Ethnic Disparities:** Minority patients may have faced systemic inequities or implicit bias in discharge planning, influencing readmission patterns in training data.
- **Geographic Variation:** Differences in hospital policies, resources, and community health infrastructure may affect readmission rates in ways unrelated to individual risk.

Impact on Patient Outcomes:

- Models may unfairly label certain populations as "high-risk," resulting in over-monitoring, stigma, or inappropriate allocation of scarce resources.
- Alternatively, they may under-predict risk for underserved groups if data under-represents their true needs, leading to missed follow-up care and worse outcomes.

Interpretation:

Biased training data can perpetuate and amplify existing health disparities, undermining the ethical goal of equitable care and harming the very patients the model is intended to help.

3.1.2 Suggest 1 Strategy to Mitigate This Bias

☑ **Fairness-Aware Reweighting and Auditing Strategy**

- **Audit Predictions:**
    - Analyze model performance across demographic groups (e.g., insurance type, age band, race if available) to identify disparities in precision, recall, and false positive/negative rates.
- **Apply Sample Weights:**
    - During training, use sample weighting to balance representation and reduce bias toward majority classes or privileged groups.

Code Example:

```
from sklearn.utils import compute_sample_weight


sample_weights = compute_sample_weight(class_weight='balanced', y=y_train)

model.fit(X_train, y_train, sample_weight=sample_weights)
```

- **Monitor Post-Deployment:**
    - Continuously evaluate fairness metrics using updated data.
    - Engage clinical and community stakeholders to review flagged cases and guide equitable policy decisions.

Interpretation:

This approach improves model fairness, ensures compliance with ethical standards, and promotes trust and adoption by clinicians and patients alike.

3.2 Trade-offs

3.2.1 Discuss the Trade-off Between Model Interpretability and Accuracy in Healthcare

Healthcare applications demand **explainable AI** to support clinician trust, accountability, and ethical decision-making. This creates a classic trade-off:

- **High Accuracy / Low Interpretability:**
  - Complex models (e.g., deep learning, large ensembles) may achieve higher predictive performance but act as black boxes.
  - Clinicians may resist adopting recommendations they cannot understand or justify to patients.
- **High Interpretability / Potentially Lower Accuracy:**
  - Simpler models (e.g., logistic regression, shallow decision trees) provide clear, human-readable rules.
  - Easier to justify in clinical contexts but may underperform in capturing complex, non-linear relationships.

Our Example:

model = GradientBoostingClassifier(n_estimators=100, max_depth=3)

- Offers **moderate interpretability** through feature importance analysis:

top10_importance_df

- However, demonstrated **low recall** in real results:

Precision: 0.47

Recall: 0.01

Interpretation:

While feature importance helps explain predictions, the model's poor sensitivity for true readmissions shows the cost of balancing interpretability and accuracy. In healthcare, the optimal solution often prioritizes interpretability even at some loss in raw accuracy to ensure safe, explainable care.

3.2.2 If the Hospital Has Limited Computational Resources, How Might This Impact Model Choice?

Resource constraints influence feasible model deployment strategies:

- **High-Complexity Models:**
  - Require more CPU/GPU resources for training and inference.
  - May increase latency or cost, making them unsuitable for smaller hospitals or clinics.
- **Simpler Models:**
  - Require less computation.
  - Easier to deploy on local servers or low-cost cloud instances.
  - May sacrifice some predictive performance but are often sufficient when designed carefully.

Example Alternative Model:

```
from sklearn.linear_model import LogisticRegression

simple_model = LogisticRegression(max_iter=500)

simple_model.fit(X_train, y_train)
```

Interpretation:
Hospitals with limited IT budgets may choose simpler models or optimize complex models to be more lightweight, ensuring accessibility without compromising essential performance.Part 4: Reflection & Workflow Diagram

4.1 Reflection

4.1.1 What Was the Most Challenging Part of the Workflow? Why?

The most challenging part of the predictive modeling workflow was **dealing with class imbalance in the target variable**. In our real hospital readmission data (e.g., diabetic_data.csv), the majority of cases were **non-readmissions**, with relatively few positive 30-day readmissions.

This imbalance led to:

- Low recall scores (e.g., 0.01), meaning the model missed many true readmissions.

- Overemphasis on predicting the majority class, sacrificing sensitivity for specificity.

Interpretation:

In healthcare, failing to identify high-risk patients can have serious consequences, such as missed follow-up care or increased morbidity. Addressing this imbalance is critical for ethical, effective deployment.

4.1.2 How Would You Improve Your Approach with More Time/Resources?

If given more time and resources, I would improve the approach in several key ways:

- **Apply Advanced Resampling Techniques:**
    - Use SMOTE (Synthetic Minority Oversampling Technique) to balance classes.
    - Experiment with ensemble methods designed for imbalance (e.g., BalancedRandomForest).
    - Example conceptual code:
    - from imblearn.over_sampling import SMOTE
    - smote = SMOTE(random_state=42)
    - X_train_res, y_train_res = smote.fit_resample(X_train, y_train)
- **Conduct Extensive Hyperparameter Tuning:**
    - Use GridSearchCV or Bayesian optimization to find optimal model parameters.
    - Emphasize maximizing recall to improve sensitivity for true readmissions.
- **Expand Feature Engineering:**
    - Incorporate time-series features (e.g., trend in hospital visits).
    - Include social determinants of health or post-discharge follow-up data.
- **Engage Domain Experts:**
    - Work closely with clinicians to validate important features.
    - Ensure model aligns with real-world care planning workflows.

Interpretation:
These improvements would produce a more balanced, sensitive, and clinically actionable model ready for deployment in real hospital settings

4.2 Diagram

4.2.1 AI Development Workflow Diagram

Below is a **simple text-based flowchart** of the AI development workflow you can **paste into Word** or **redraw with shapes**:

```
+-------------------------+
| 1. Problem Definition    |
| - Define goals           |
| - Identify stakeholders  |
+----------↓-------------+
| 2. Data Collection       |
| - Gather EHR, demographics|
| - Ensure privacy         |
+----------↓-------------+
| 3. Data Preprocessing    |
| - Clean missing data     |
| - Encode variables       |
| - Normalize features     |
+----------↓-------------+
| 4. Model Development     |
| - Choose model           |
| - Train & tune           |
| - Validate on split data |
+----------↓-------------+
| 5. Evaluation            |
| - Confusion matrix       |
| - Precision & recall     |
| - F1-score               |
```

```
+-----------↓--------------+
| 6. Deployment            |
| - API integration        |
| - EHR system connection  |
| - User training          |
+-----------↓--------------+
| 7. Monitoring & Updates  |
| - Check for drift        |
| - Retrain as needed      |
| - Fairness audits        |
+--------------------------+
```

4.3 Interpretation

This workflow represents a complete, iterative AI development process tailored for healthcare settings. It emphasizes **data quality**, **clinical validation**, and **continuous monitoring**, ensuring ethical and effective deployment in hospital environments.