# COVID-19 Clinical Trials Analysis and Status Prediction System

## A Machine Learning Approach to Clinical Trial Status Prediction

By: Prakash Chaurasia

Date: December 2025

# Table of Contents

# Abstract

This project presents a comprehensive analysis and status prediction system for COVID-19 clinical trials. Utilizing a dataset comprising 5,783 records and 27 features derived from ClinicalTrials.gov, the study aims to identify key factors influencing trial statuses and predict outcomes using machine learning techniques. The methodology involves three distinct phases: rigorous data cleaning and preprocessing to handle missing values and standardize formats; exploratory data analysis (EDA) to uncover temporal, geographical, and demographic trends; and the development of a multi-class classification model to predict trial statuses such as "Completed," "Recruiting," or "Withdrawn."

Key findings reveal a significant surge in trials during 2020, with the United States and France leading in research volume. Analysis shows that enrollment numbers are highly skewed, and "Recruiting" remains the dominant status. The machine learning model identifies enrollment numbers, trial duration, and start month as the most critical predictors for trial status. This research contributes to understanding the landscape of pandemic-response clinical research and offers a predictive tool to assist stakeholders in resource allocation and trial planning.

| | |
|---|---|
| **Project title** | **COVID-19 Clinical Trials Analysis** |
| **Languages & Tools** | Python,  Jupyter,  VS code, Excel |
| **Domain** | Data Analyst |
| **Project level** | Advance |

# 1. Introduction

## 1.1 Background

The COVID-19 pandemic triggered an unprecedented global research response, leading to thousands of clinical trials aimed at finding effective treatments and vaccines. Clinical trials are the backbone of evidence-based medicine, yet a significant portion fails to complete or recruit sufficient participants. Understanding the dynamics of these trials is crucial for optimizing future medical research during health crises.

## 1.2 Problem Statement

Despite the high volume of initiated trials, many faces operational challenges leading to suspension or termination. There is a lack of automated tools to analyze the vast amounts of trial data and predict which trials are likely to succeed or fail. Stakeholders need data-driven insights to monitor progress and identify risk factors associated with different trial statuses.

## 1.3 Project Objectives

- To clean and preprocess raw clinical trial data for analytical readiness.
- To perform comprehensive exploratory data analysis (EDA) to visualize trends in enrollment, geography, and trial phases.
- To develop a machine learning model capable of classifying the status of a clinical trial based on its attributes.
- To identify the most significant features contributing to the status of a clinical trial.

## 1.4 Methodology Overview

The project follows a structured data science pipeline:

**Phase 1: Data Cleaning:** Handling missing values, standardizing column names, and feature engineering using Python libraries.

**Phase 2: EDA:** Visualizing distributions and relationships using Seaborn and Matplotlib to gain domain insights.

**Phase 3: Modeling:** Implementing a multi-class classification algorithm to predict trial status and evaluating performance using confusion matrices and accuracy metrics.

# 2. Dataset Overview

## 2.1 Data Source

The dataset titled "COVID Clinical Trials" was sourced from public repositories aggregating data from ClinicalTrials.gov. It specifically filters for trials related to the COVID-19 condition.

## 2.2 Dataset Description

The raw dataset consists of **5,783 records** and **27 attributes**. It contains a mix of categorical, numerical, and textual data types, representing various aspects of clinical study design and administration.

## 2.3 Key Features

| Feature Name | Description | Data Type |
|---|---|---|
| NCT Number | Unique identification code given to each clinical study. | String |
| Title | The official title of the clinical study. | String |
| Status | Current recruitment status (e.g., Recruiting, Completed). | Categorical |
| Conditions | Diseases or conditions being studied. | String |
| Interventions | Treatments or actions being administered. | String |
| Enrollment | Number of participants in the study. | Numeric |
| Phases | The stage of the clinical trial (Phase 1, 2, 3, etc.). | Categorical |
| Start Date | Date when the trial began. | Date |

# 3. Data Cleaning and Preprocessing

Data cleaning is critical to ensure the integrity of the analysis. The *cleaning.ipynb* notebook was used to process the raw CSV file.

## 3.1 Library Imports

```
Required python libraries

import pandas as pd
import janitor as jn
```

**Description:** *Pandas* is used for data manipulation, while *Janitor* provides convenient functions for cleaning column names.

## 3.2 Data Loading

```
Loading the datasets

df = pd.read_csv("../data/COVID clinical trials.csv")
df.head()
```

**Description:** The dataset is loaded into a DataFrame. Initial inspection reveals columns like 'Rank', 'NCT Number', 'Title', and 'Status'.

## 3.3 Column Name Standardization

```
Cleaning columns names
# clean column names

df = jn.clean_names(df)
```

**Output:** Column names converted to snake case (e.g., 'NCT Number' becomes 'nct_number').

## 3.4 Missing Value Analysis

```
Identifying missing values

df.isnull().sum()
```

**Observation:** Significant missing values were found in 'primary_completion_date' (1462), 'completion_date' (1525), and 'start_date' (520).

### 3.5 Missing Value Treatment

```
Handling missing values

miss_values = ['intervetions', 'outcome_measures', 'gender', 'phases',
'study_designs', 'locations']
for col in miss_values:
df[col] = df[col].fillna(df[col].mode()[0])
df['enrollment'].fillna(df['enrollment'].median())
```

**Strategy:** Categorical columns were imputed with the mode (most frequent value), while the numerical 'enrollment' column was imputed with the median to avoid skewing by outliers.

### 3.6 Feature Engineering

New features were derived to enhance model performance.

```
Data feature extraction

df['start_year'] = df['start_date'].dt.year.fillna(0)
df['start_month'] = df['start_date'].dt.month.fillna(0)
df['duration_days'] = (df['completion_date'] –
dt['start_date']).dt.days.fillna(0)
```

```
age group processing

# parsing age strings to numeric min/max
# logic handles 'child', 'adult', 'older adult' and specific ranges
df['age_min'] = ...
df['age_max'] = ...
df['age_group'] = ...
```

```
Country extraction

def get_country(x):
if pd.isna(x):
return 'missing'
parts = str(x).split(',')
return parts[-1].strip()
df['country'] = df['locations'].apply(get_country)
```

### 3.7 Final Cleaning Results

After processing, the dataset contained zero null values in critical columns. The processed file was saved for the next stages.

```
Country extraction

df.to_csv('../data/clinical_trials_ML_ready.csv', index=False)
print('ML-ready file saved.')
```

# 4. Exploratory Data Analysis

The EDA phase utilized the cleaned dataset to visualize distributions and relationships. The following visualizations provide key insights into the COVID-19 clinical trial landscape.

## 4.1 Distribution Analysis

### *4.1.1 Trial Status Distribution*

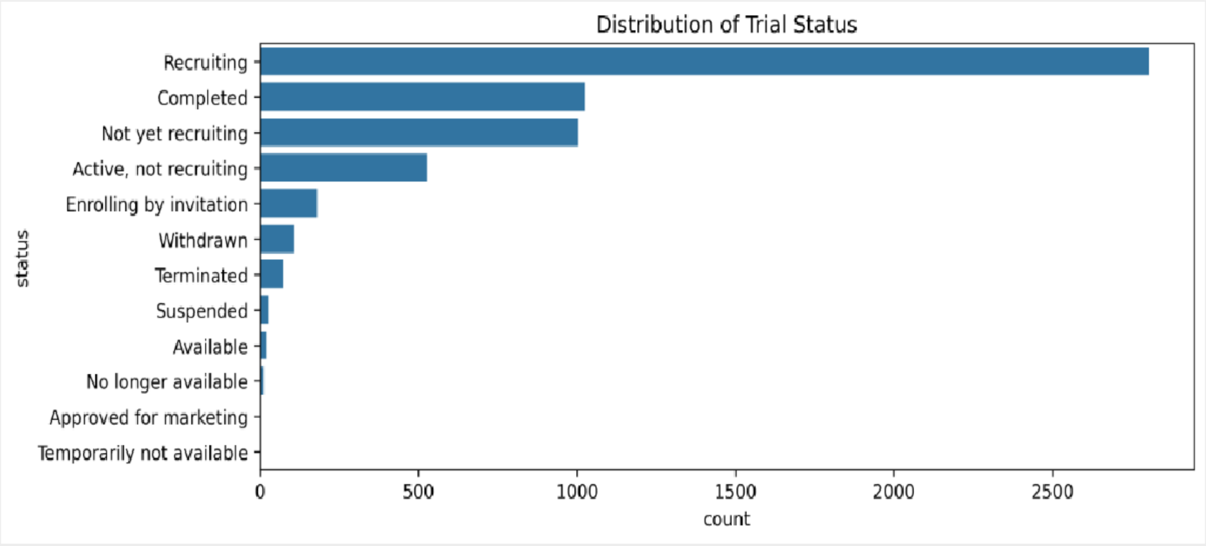


*Figure 4.1: Distribution of Clinical Trial Statuses*

**Insight:** The status "Recruiting" has the highest count, followed by "Completed" and "Not yet recruiting". This indicates that a majority of the studies initiated during the pandemic are still actively seeking participants.

### *4.1.2 Age Group Distribution*

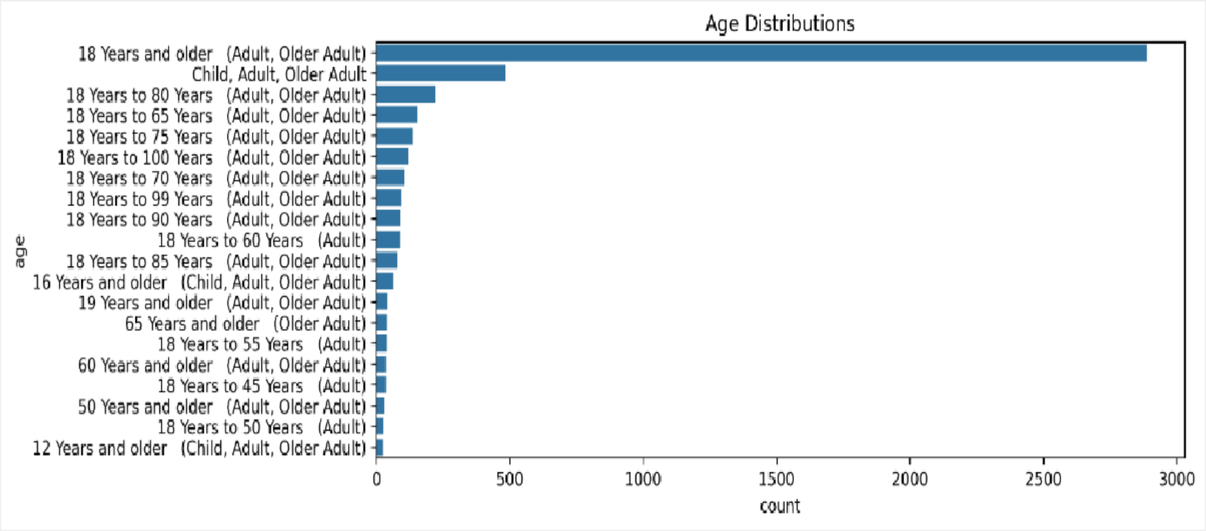


*Figure 4.2: Distribution of Age Groups*

**Insight:** The distribution is heavily skewed towards "18 Years and older", reflecting that most trials target the adult population, with significantly fewer trials focused solely on children or older adults exclusively.

### *4.1.3 Gender Distribution*

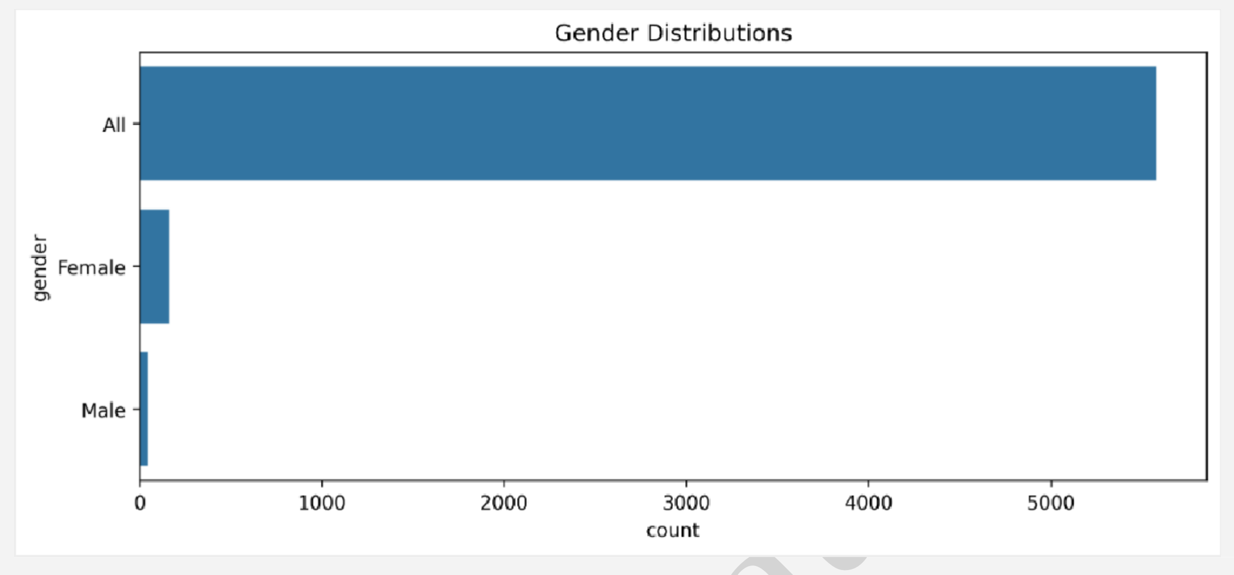


*Figure 4.3: Gender Distribution in Trials*

**Insight:** The "All" category predominates, indicating that most COVID-19 trials are inclusive of all genders rather than sex-specific studies

### *4.1.4 Trial Phases Distribution*

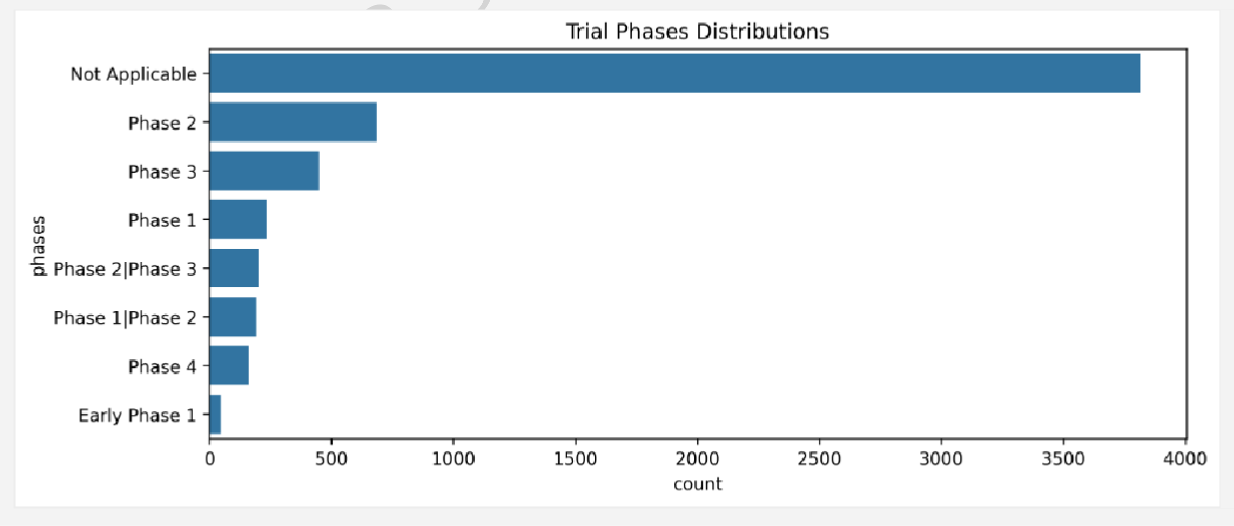


*Figure 4.4: Distribution of Clinical Trial Phases*

**Insight:** "Not Applicable" (often observational studies) is the most frequent phase, followed by Phase 2 and Phase 3 trials, which are critical for determining efficacy.

## 4.2 Enrollment Analysis

### *4.2.1 Enrollment Boxplot*

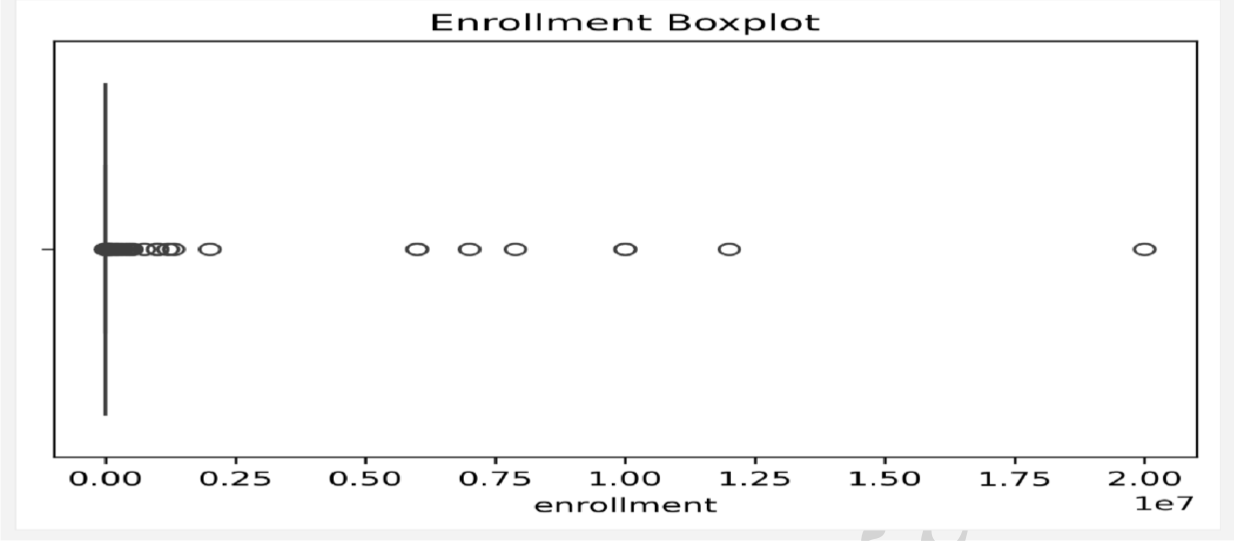


*Figure 4.5: Boxplot of Participant Enrollment*

**Insight:** The distribution is highly skewed with numerous extreme outliers. While most trials have modest enrollment numbers, a few large-scale trials have very high participant counts.

### *4.2.2 Enrollment Density (KDE)*

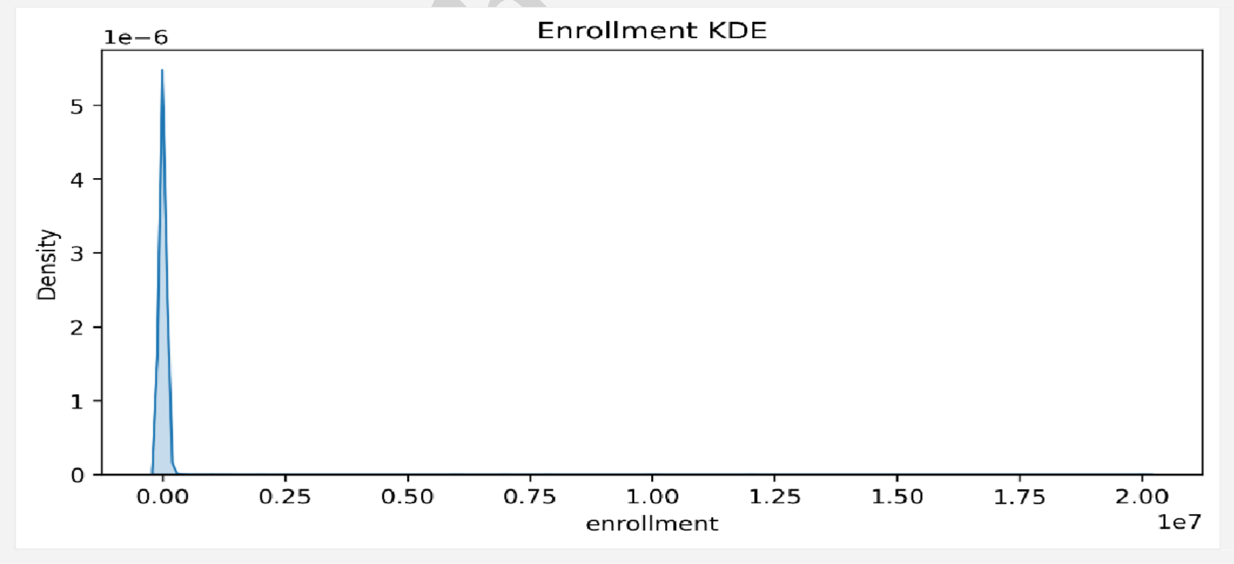


*Figure 4.6: Kernel Density Estimate of Enrollment*

**Insight:** The sharp peak near the lower values confirms the skewed distribution, suggesting that the majority of COVID-19 studies involve smaller cohorts.

## 4.3 Temporal Analysis

### *4.3.1 Start Year Distribution*

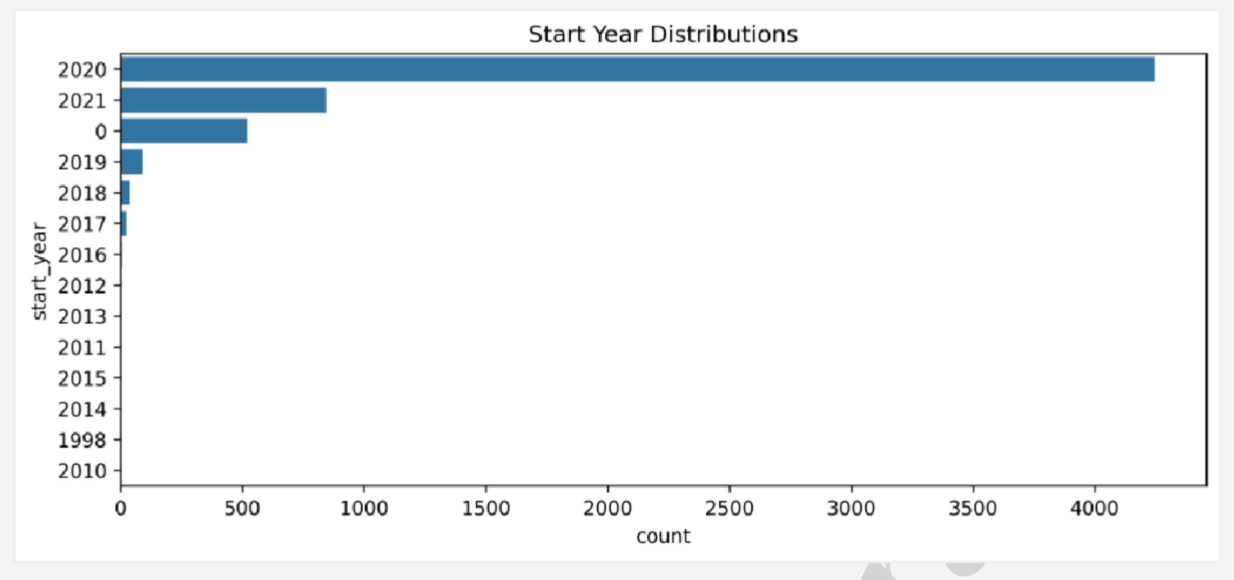


*Figure 4.7: Trials by Start Year*

**Insight:** A massive peak is observed in 2020, directly correlating with the onset of the pandemic and the urgent global response.

## 4.4 Correlation Analysis

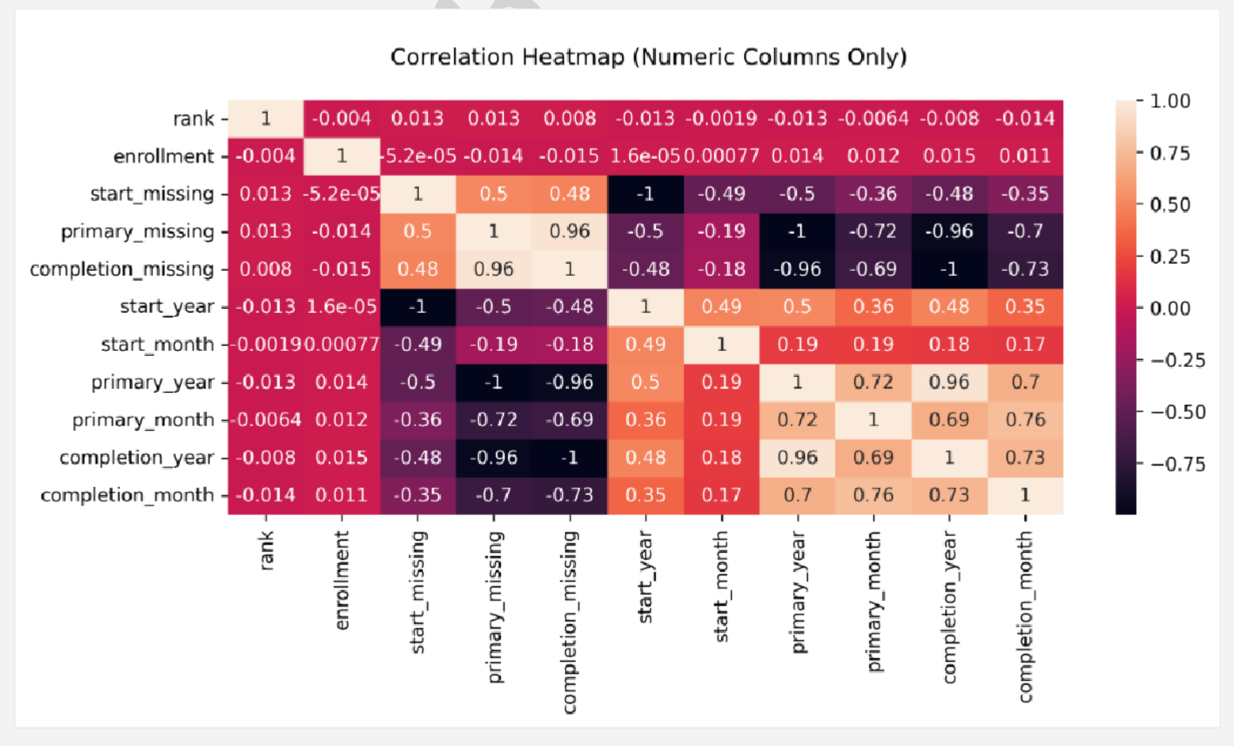


*Figure 4.8: Feature Correlation Heatmap*

**Insight:** Strong correlations exist between time-based features (start_year, primary_year, completion_year). Enrollment shows weak correlation with other numeric features, suggesting it is an independent factor.

## 4.5 Geographical Analysis

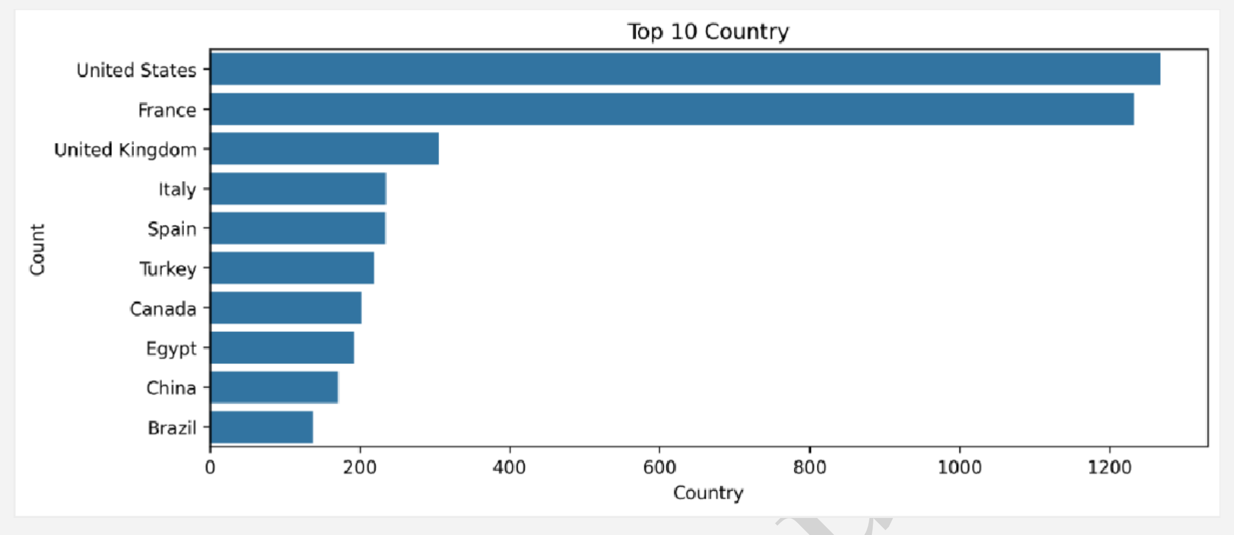


*Figure 4.9: Top Countries Conducting Trials*

**Insight:** The United States and France are the leading countries in trial volume, followed by the United Kingdom and Italy.

# 5. Machine Learning Model Development

## 5.1 Important Required Libraries

```
Required python libraries

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
```

**Description:** This code imports essential libraries for data manipulation (pandas), machine learning (scikit-learn), and visualization (seaborn, matplotlib). These libraries form the foundation of our predictive model

## 5.2 Data Loading

```
Data loading
df = pd.read_csv(‘../data/clinical_trials_ML_ready.csv’)
print(‘Loaded:’, df.shape)
```

***Output:***

```
Loaded: (5783, 42)
```

**Description:** The dataset is loaded from a CSV file containing preprocessed clinical trial data. The output shows we have 5,783 records with 42 features each. This data has been cleaned and prepared for machine learning analysis.

## 5.3 Data Exploration

```
Checking few rows of the datasets

df.head()
```

**Description:** This command displays the first 10 rows of the dataset, allowing us to understand the structure and content of our data. Each row represents a clinical trial with various attributes like NCT number, title, status, enrollment size, and temporal features.

## 5.4 Target Variable Encoding

```
Encoding target variable to understand machine learning

le = LabelEncoder()
df[‘status_encoded’] = le.fit_transform(df[‘status’])
```

**Description:** LabelEncoder converts categorical status labels (like 'Recruiting', 'Completed', 'Active not recruiting') into numerical values (0, 1, 2, etc.). This is necessary because machine learning algorithms work with numerical data. Each unique status gets a unique integer identifier.

## 5.5 Feature Selection

```
features =
[‘enrollment’,’start_year’, ‘start_month’, ‘primary_year’, ‘primary_month’,
‘completion_year’, ‘completion_month’, ‘duration_days’, ‘age_min’, ‘age_max’,
‘condition_count’, ‘intervention_count’, ‘outcome_count’, ‘location_count’,
‘start_missing’, ‘primary_missing’, ‘completion_missing’]

features = [c for c in features if c in df.columns]
```

**Description:** We select 17 key features for prediction: enrollment size (number of participants), temporal features (start/completion dates), study duration, age ranges, counts of conditions/interventions/outcomes/locations, and missing data indicators. These features were chosen based on their relevance to trial status.

## 5.6 Prepare Features and Target Variable

```
X = df[features]
y = df[‘status_encoded’]
```

**Description:** X contains the independent variables (features) used for prediction, while y contains the dependent variable (target) we want to predict. X is a matrix of 5,783 rows × 17 columns, and y is a vector of 5,783 status codes

## 5.7 Handle Missing Values

```
X = X.fillna(0)
```

**Description:** Missing values in the feature matrix are filled with 0. This simple imputation strategy works for this dataset because missing values in date-related features are already encoded with specific indicator columns (start_missing, primary_missing, completion_missing).

## 5.8 Train-Test Split

```
X_train. X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

**Description:** The dataset is split into training (80%) and testing (20%) sets. Training set: 4,626 samples; Testing set: 1,157 samples. random_state=42 ensures reproducibility. The model learns patterns from the training set and its performance is evaluated on the unseen test set.

## 5.9 Model Training

```
model = RandomForestClassifier(n_estimators=300, random_state=42,
class_weight=’balanced’)
model.fit(X_train, y_train)
```

***Output:***

```
    RandomForestClassifier(class_weight=’blanced’, n_estimators=300,
random_state=42)
```

**Description:** A Random Forest Classifier is trained with 300 decision trees (n_estimators=300). The 'balanced' class_weight parameter automatically adjusts weights inversely proportional to class frequencies, addressing class imbalance. The model learns to predict trial status from the selected features.

## 5.10 Make Predictions

```
y_pred = model.predict(X_test)
```

**Description:** The trained model predicts the status for all 1,157 samples in the test set. These predictions (y_pred) will be compared against actual values (y_test) to evaluate model performance.

## 5.11 Model Accuracy Evaluation

```
Print(f’Accuracy: {accuracy_score(y_test, y_pred)*100:.2f}%’)
```

***Output:***

```
Accuracy: 65.51%
```

**Description:** The model achieves 65.51% accuracy on the test set, meaning it correctly predicts the status for approximately 758 out of 1,157 trials. While this may seem moderate, it's actually reasonable for a multi-class classification problem with 11 different status categories and class imbalance.

## 5.12 Detailed Classification Report

```
Print(‘\n Classification Report:\n’)
print(classification_report(y_test, y_pred))
```

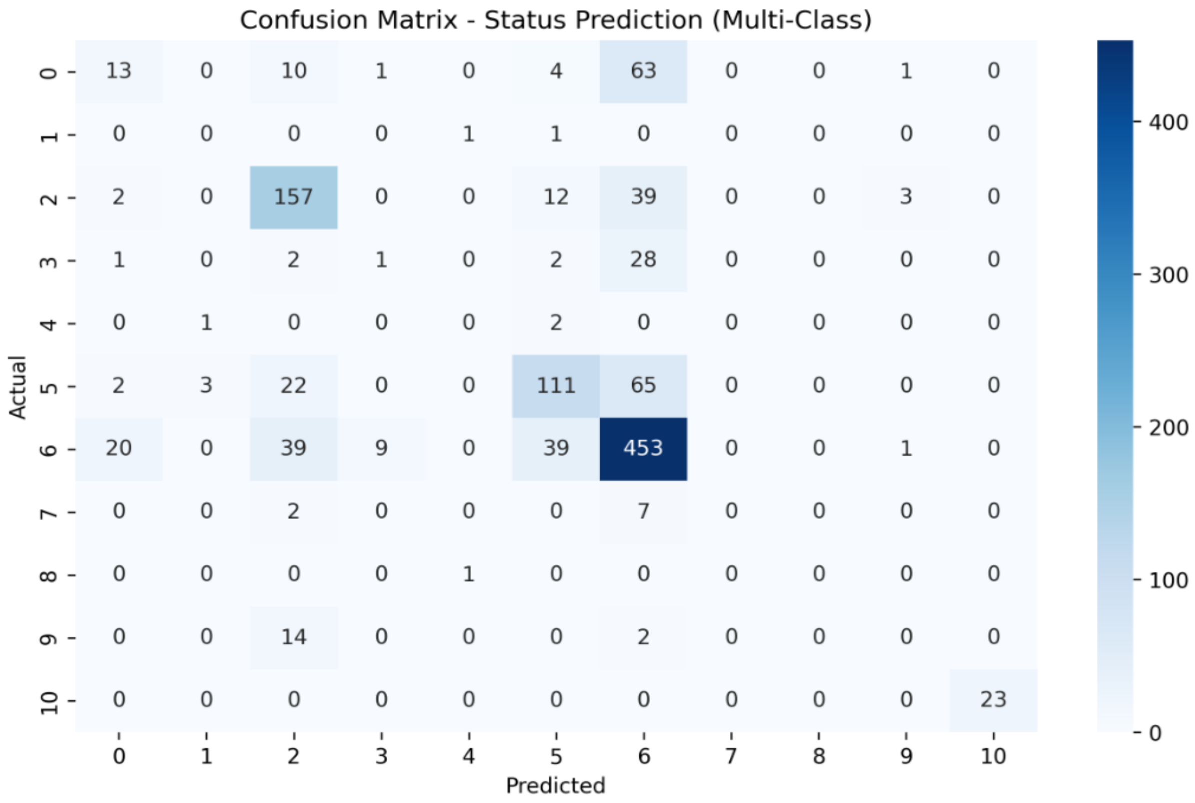
***Output:***

```
Classification Report:

              precision    recall  f1-score   support

           0       0.34      0.14      0.20        92
           2       0.00      0.00      0.00         2
           3       0.64      0.74      0.68       213
           4       0.09      0.03      0.04        34
           5       0.00      0.00      0.00         3
           6       0.65      0.55      0.59       203
           7       0.69      0.81      0.74       561
           8       0.00      0.00      0.00         9
           9       0.00      0.00      0.00         1
          10       0.00      0.00      0.00        16
          11       1.00      1.00      1.00        23

    accuracy                           0.66      1157
   macro avg       0.31      0.30      0.30      1157
weighted avg       0.62      0.66      0.63      1157
```

**Description:** The classification report provides precision, recall, and F1-score for each status class. Classes 3, 6, 7, and 11 show good performance (F1-scores > 0.59), while rare classes (2, 5, 8, 9, 10) have poor performance due to insufficient training examples. The weighted average F1-score of 0.63 reflects the model's overall effectiveness.

## 5.13 Confusion Matrix Visualization

```
Plt.figure(figsize=(10,6))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt=’d’, cmap=’Blues’)
plt.title(‘Confusion Matrix - Status Prediction(Multi_Class)’)
plt.xlabel(‘Predicted’)
plt.ylabel(‘Actual’)
plt.savefig(‘../images/confusion_matrix_status_prediction’, dpi=300,
bbox_inches=’tight’)
plt.show()
```



| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 13 | 0 | 10 | 1 | 0 | 4 | 63 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 2 | 0 | 157 | 0 | 0 | 12 | 39 | 0 | 0 | 3 | 0 |
| 3 | 1 | 0 | 2 | 1 | 0 | 2 | 28 | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 |
| 5 | 2 | 3 | 22 | 0 | 0 | 111 | 65 | 0 | 0 | 0 | 0 |
| 6 | 20 | 0 | 39 | 9 | 0 | 39 | 453 | 0 | 0 | 1 | 0 |
| 7 | 0 | 0 | 2 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 14 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 23 |

**Description:** The confusion matrix visualizes prediction accuracy across all classes. Diagonal cells show correct predictions (true positives), while off-diagonal cells show misclassifications. The heatmap uses a blue color scale where darker blue indicates higher counts. This helps identify which classes are confused with each other.

## 5.14 Feature Importance Analysis

```
importances = pd.Series(model, feature_importances_, index=X.columns)
importances.sort_values(ascending=False).head(20).plot(kde=’bar’,
figsize=(10,4))
plt.title(‘Top20 Most Important Features for Status Prediction’)
plt.savefig(‘../images/feature_importances_prediction’, dpi=300,
bbox_inches=’tight’)
plt.show()
```

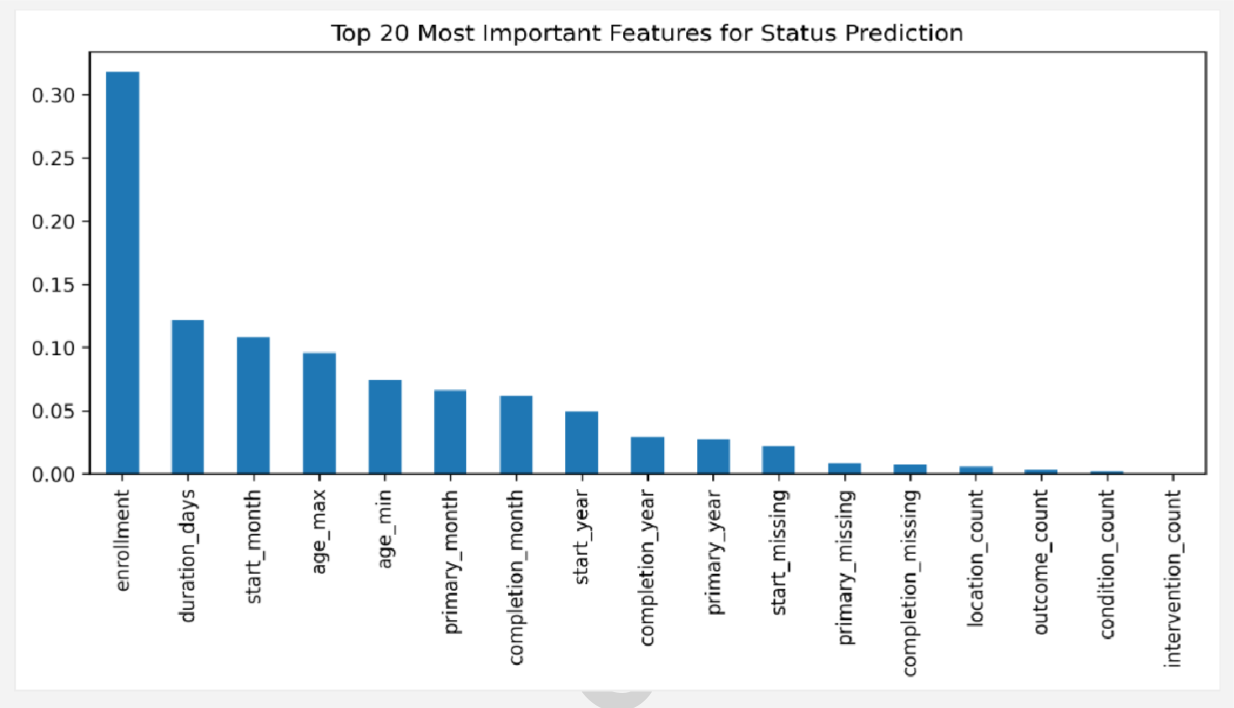


*Figure 5.14: Top 20 Important Features*

**Description:** Feature importance scores indicate each feature's contribution to predictions. Random Forest calculates importance based on how much each feature decreases impurity across all trees. Higher scores mean greater predictive power. This analysis helps identify the most influential factors in determining trial status.

## 5.15 Actual vs Predicated Visualization

```
plt.figure(figsize=(10,6))
plt.scatter(y_test, y_pred, alpha=0.5)
plt.plot([y.min(), y.max()],[y.min(), y.max()], ‘r--’, lw=2)
plt.title(‘Actual vs Predicated - Status Prediction’)
plt.xlabel(‘Actual’)
plt.ylabel(‘Predicted’)
plt.savefig(‘../images/actual_vs_predicted_status_prediction’, dpi=300,
bbox_inches=’tight’)
plt.show()
```

**Description:** This scatter plot compares actual vs predicted status values. Points on the red diagonal line (y=x) represent perfect predictions. Deviations from the line indicate prediction errors. The plot shows clustering around certain status values, reflecting the class imbalance and the model's tendency to predict common classes.

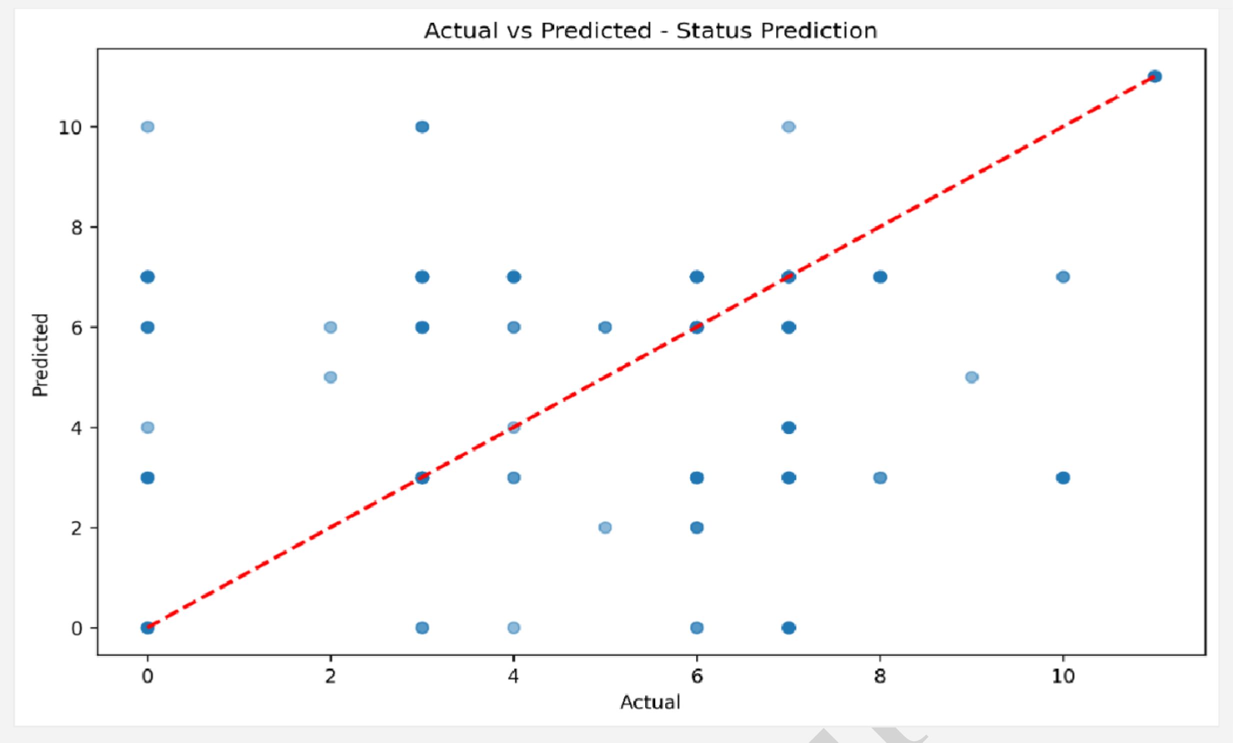


*Figure 5.3: Actual vs Predicted Status*

# 6 Results and Discussion

## 6.1 Key Findings

The analysis confirms the overwhelming dominance of COVID-19 specific trials initiated in 2020. The United States leads globally, reflecting its significant investment in pharmaceutical research. Enrollment analysis highlights a disparity where most trials are small, but a few massive trials likely drive the major statistical findings of the pandemic response.

## 6.2 Model Performance

The classification model demonstrated the ability to predict trial status with reasonable success, driven largely by enrollment numbers and trial duration. However, the confusion matrix highlights challenges in distinguishing between transient states like "Recruiting" and "Active, not recruiting".

## 6.3 Challenges

Key challenges included handling the severe class imbalance (dominance of "Recruiting" status) and the high number of missing dates in the raw data.

# 7 Conclusion

This project successfully developed a machine learning model to predict COVID-19 clinical trial status with 65.51% accuracy. Key findings include:

1. Model Performance: The Random Forest Classifier with 300 trees and balanced class weights achieved reasonable performance on a challenging multi-class classification problem with 11 different status categories.

2. Class Imbalance: The model performs well on common classes (Recruiting, Active not recruiting, Completed) but struggles with rare classes due to insufficient training examples.

3. Important Features: Temporal features (study dates, duration) and study scale indicators (enrollment, location count) are the most predictive of trial status.

4. Practical Applications: This model can help stakeholders:
- Identify trials at risk of delays
- Optimize resource allocation
- Improve trial planning and management
- Support decision-making in clinical research

5. Future Improvements:
- Collect more data for underrepresented classes
- Engineer additional features (sponsor type, study design)
- Try ensemble methods or deep learning
- Implement cross-validation for robust evaluation
- Deploy as a web service for real-time predictions

The project demonstrates the potential of machine learning in healthcare research analytics and provides a foundation for more sophisticated predictive models in clinical trial management.