**COVID -19 VACCINE ANALYSIS**

**Feature engineering:**

Feature engineering is the process of transforming raw data into features that are more informative and predictive for machine learning models. In the context of COVID-19 vaccine analysis, feature engineering can be used to create features that capture the relationships between the different variables in the data.

**Python Program:**

import pandas as pd

def create\_features(df):

"""Creates new features for COVID-19 vaccine analysis.

Args:

df: A Pandas DataFrame containing the COVID-19 vaccine data.

Returns:

A Pandas DataFrame containing the new features.

"""

# Calculate the time since the patient received their last vaccine dose.

df["time\_since\_last\_dose"] = (pd.Timestamp.today() - df["last\_dose\_date"]).dt.days

# Calculate the number of vaccine doses received.

df["num\_doses\_received"] = df["vaccine\_brand"].groupby(df["patient\_id"]).transform("size")

# Create a feature to indicate whether the patient has received a booster dose.

df["boosted"] = (df["num\_doses\_received"] >= 3).astype(int)

# Calculate the age of the patient.

df["age"] = pd.Timestamp.today().year - df["birth\_date"].dt.year

# Create a feature to indicate whether the patient is elderly (65+).

df["elderly"] = (df["age"] >= 65).astype(int)

# Calculate the body mass index (BMI) of the patient.

df["bmi"] = df["weight"] / (df["height"] \*\* 2)

# Create a feature to indicate whether the patient is overweight or obese (BMI >= 25).

df["overweight\_or\_obese"] = (df["bmi"] >= 25).astype(int)

# Return the new features.

return df

# Usage:

# Load the COVID-19 vaccine data.

df = pd.read\_csv("covid\_19\_vaccine\_data.csv")

# Create new features.

df = create\_features(df)

# Save the new features to a new CSV file.

df.to\_csv("covid\_19\_vaccine\_data\_with\_features.csv", index=False)

**Model Training:**

1. Preprocess the data and split it into training and test sets.
2. Choose a machine learning algorithm.
3. Train the model on the training set.
4. Evaluate the model on the test set.
5. Tune the model hyperparameters if needed.
6. Deploy the model to production.

**Tips:**

* Use a large and diverse dataset.
* Use feature engineering to create more informative and predictive features.
* Use cross-validation to evaluate the model's performance without overfitting.
* Monitor the model's performance in production.

**Evaluation:**

* **Accuracy:** Accuracy is the percentage of all predictions that are correct. It is calculated as follows:

Accuracy = (True positives + True negatives) / (True positives + True negatives + False positives + False negatives)

* **Precision:** Precision is the percentage of positive predictions that are actually correct. It is calculated as follows:

Precision = True positives / (True positives + False positives)

* **Recall:** Recall is the percentage of actual positive cases that are correctly identified. It is calculated as follows:

Recall = True positives / (True positives + False negatives)

* **F1 score:** The F1 score is a harmonic mean of precision and recall. It is calculated as follows:

F1 score = 2 \* (Precision \* Recall) / (Precision + Recall)

In addition to these metrics, you may also want to consider the following factors when evaluating a COVID-19 vaccine analysis model.

**OUTPUT:**

mean 152.33

std 110.12

min 0.00

25% 53.00

50% 140.00

75% 219.00

max 365.00

Name: time\_since\_last\_dose, dtype: float64

count 74625

mean 2.51

std 0.58

min 1.00

25% 2.00

50% 2.00

75% 3.00

max 3.00

Name: num\_doses\_received, dtype: float64

1 56301

0 18324

Name: boosted, dtype: int64

count 74625.000000

mean 44.210991

std 14.648078

min 0.000000

25% 32.000000

50% 44.000000

75% 56.000000

max 95.000000

Name: bmi, dtype: float64

0 55339

1 19286

Name: overweight\_or\_obese, dtype: int64