Purwadhika
Digital Technology School

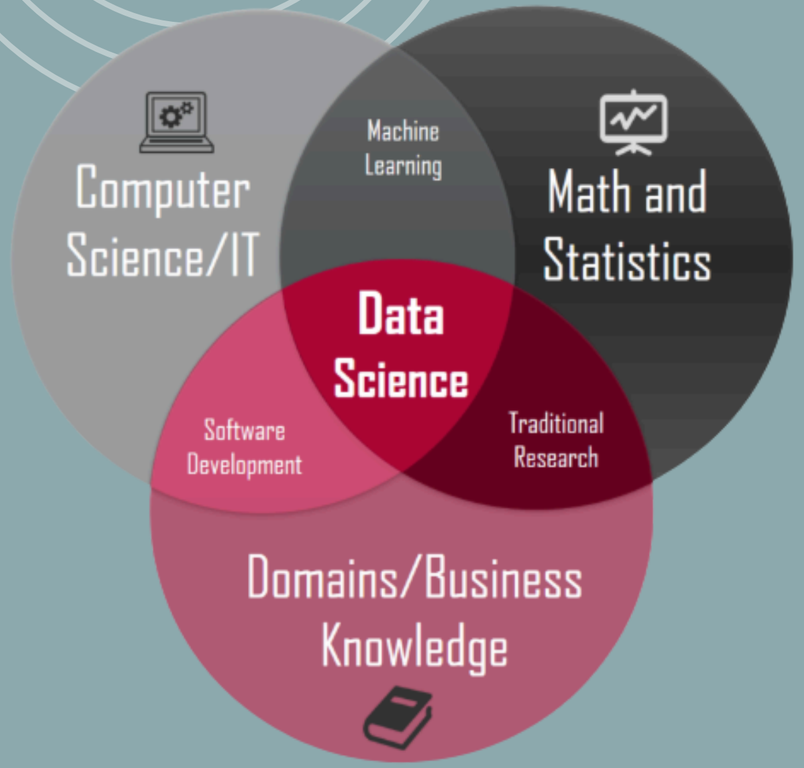
# DATA SCIENTIST WORK FLOW

*Harish Muhammad*

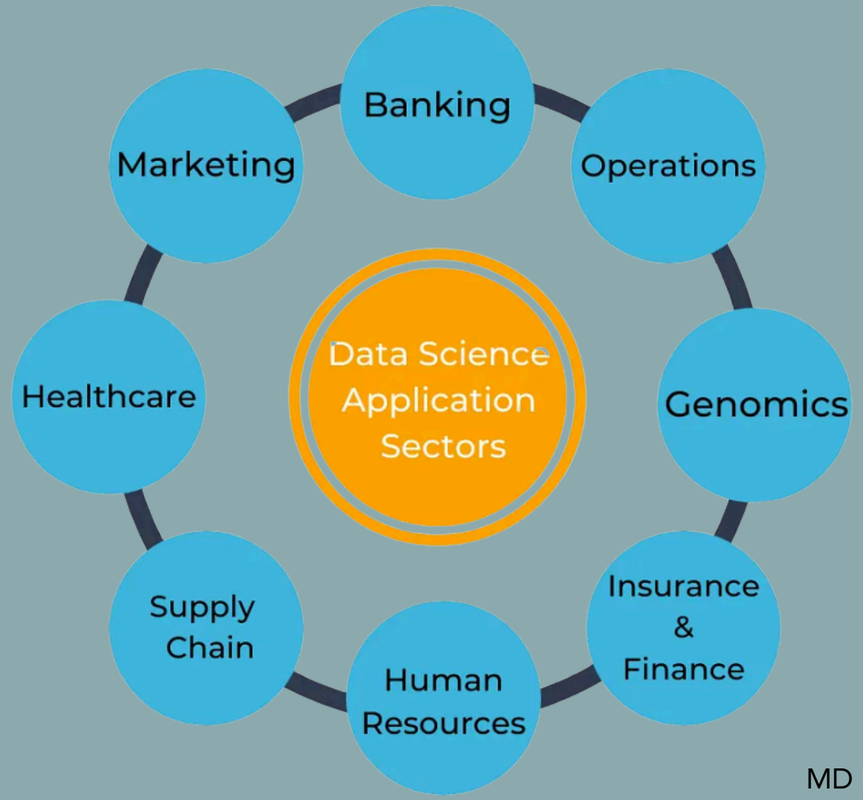
# OUTLINE PRESENTASI

01. APA ITU DATA SCIENCE?
02. ALUR KERJA DATA SAINS
03. PEMAHAMAN BISNIS
04. DATA UNDERSTANDING
05. DATA PREPARATION
06. PEMODELAN MACHINE LEARNING
07. EVALUASI
08. MODEL DEPLYOMENT
09. KESIMPULAN

# APA ITU DATA SAINS?



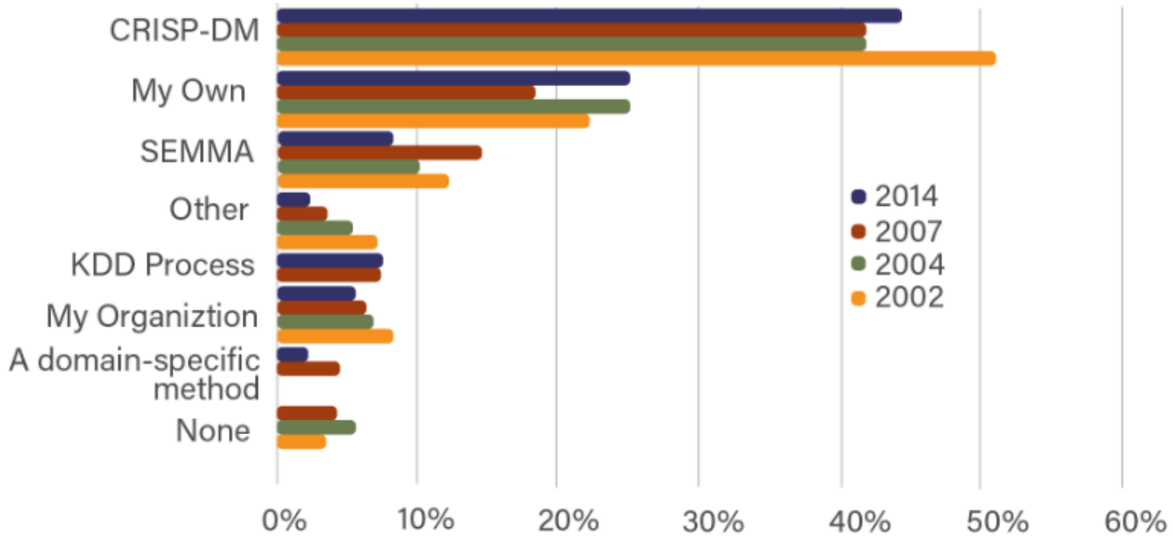

thedatascientist.com




MD Irfan

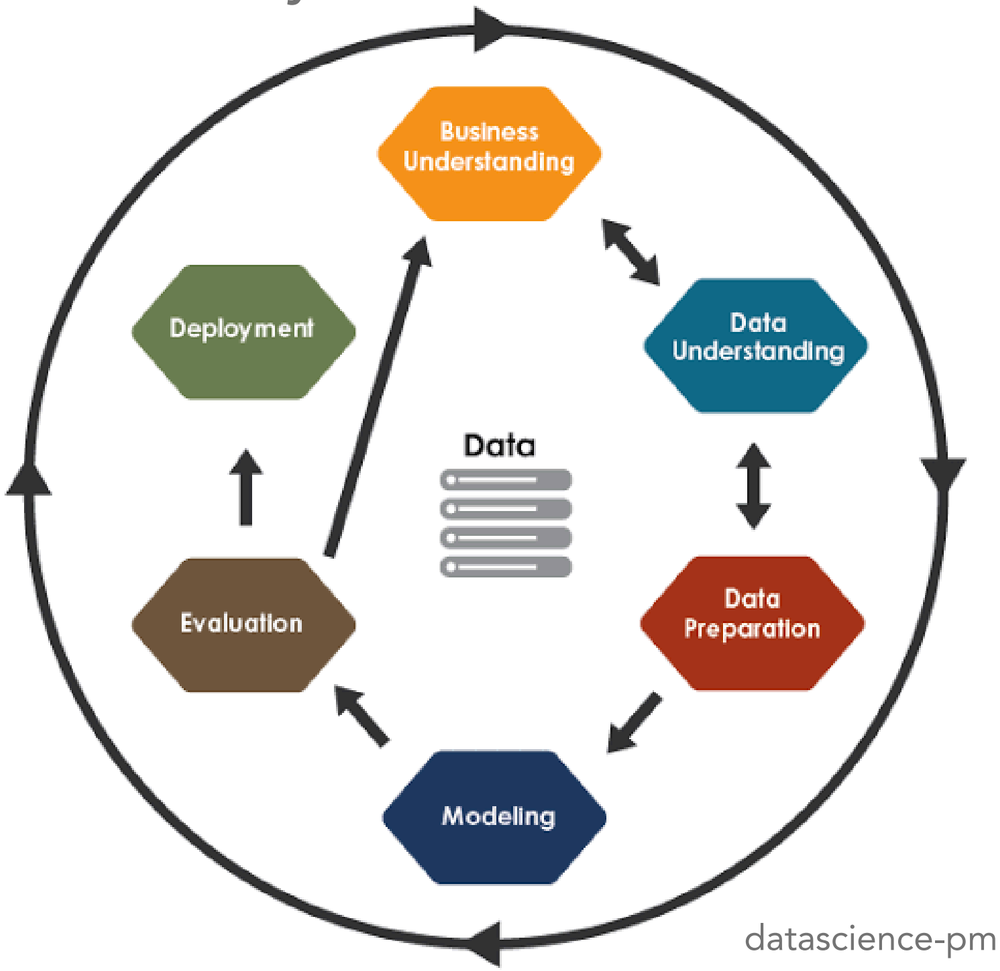
# ALUR KERJA BERDASARKAN CRISP-DM

## Popularitas Penggunaan CRISP-DM



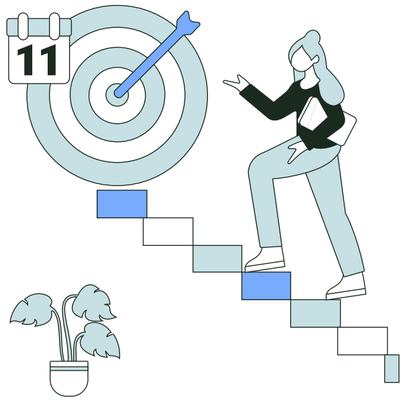

KDnuggets polls

## Cross Industry standard for data mining




datascience-pm.com

# PEMAHAMAN BISNIS (BUSINESS UNDERSTANDING)




- Memahami orientasi & kebutuhan bisnis
- Berdiskusi dengan stakeholders
- Melihat situasi & ketersediaan sumber daya
- Menentukan tujuan proyek DS
- Perencanaan proyek DS

# PEMAHAMAN DATA (DATA UNDERSTANDING)

## Pengumpulan data

- Sumber data
- Lokasi
- Metode
- Unsur privasi

## Data Description

- Tipe data
- Volume data

## Explorasi Data

- Visualisasi data
- Statistik deskriptif
- Trend data
- Korelasi

# Pengumpulan data, deskripsi dan explorasi data

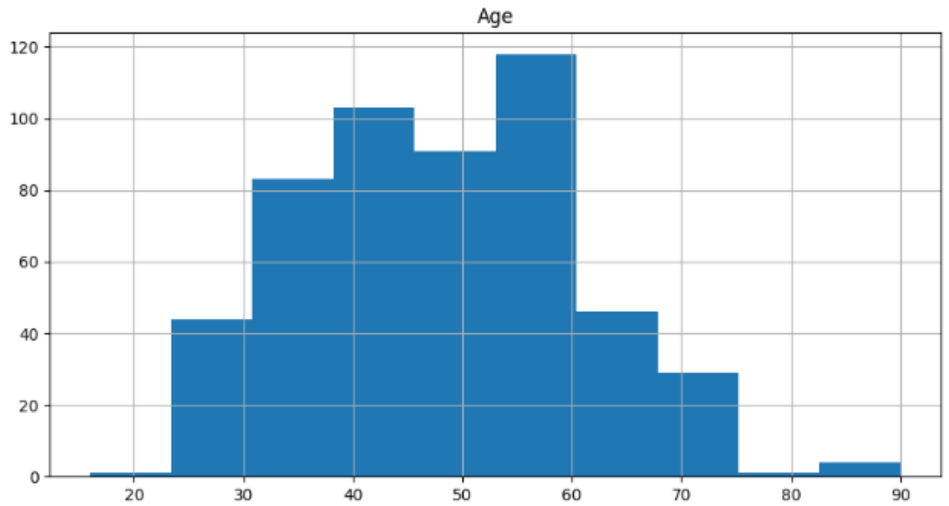
```
import pandas as pd
file_name = 'diabetes_data.csv'
df = pd.read_csv(file_name)
display(df.info(), df.describe(include='O').T)
```

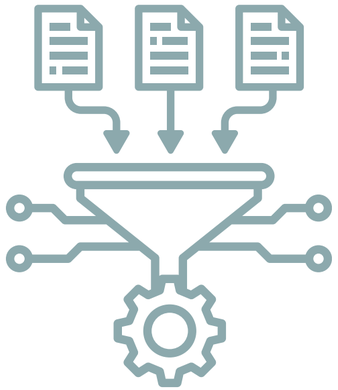
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 520 entries, 0 to 519
Data columns (total 17 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   Age                 520 non-null    int64 
 1   Gender              520 non-null    object
 2   Polyuria            520 non-null    object
 3   Polydipsia          520 non-null    object
 4   sudden weight loss  520 non-null    object
 5   weakness            520 non-null    object
 6   Polyphagia          520 non-null    object
 7   Genital thrush      520 non-null    object
 8   visual blurring     520 non-null    object
 9   Itching             520 non-null    object
 10  Irritability        520 non-null    object
 11  delayed healing     520 non-null    object
 12  partial paresis     520 non-null    object
 13  muscle stiffness    520 non-null    object
 14  Alopecia            520 non-null    object
 15  Obesity             520 non-null    object
 16  class               520 non-null    object
dtypes: int64(1), object(16)
memory usage: 69.2+ KB
None
```

| | count | unique | top | freq |
|---|---|---|---|---|
| **Gender** | 520 | 2 | Male | 328 |
| **Polyuria** | 520 | 2 | No | 262 |
| **Polydipsia** | 520 | 2 | No | 287 |
| **sudden weight loss** | 520 | 2 | No | 303 |
| **weakness** | 520 | 2 | Yes | 305 |
| **Polyphagia** | 520 | 2 | No | 283 |
| **Genital thrush** | 520 | 2 | No | 404 |
| **visual blurring** | 520 | 2 | No | 287 |
| **Itching** | 520 | 2 | No | 267 |
| **Irritability** | 520 | 2 | No | 394 |
| **delayed healing** | 520 | 2 | No | 281 |
| **partial paresis** | 520 | 2 | No | 296 |
| **muscle stiffness** | 520 | 2 | No | 325 |
| **Alopecia** | 520 | 2 | No | 341 |
| **Obesity** | 520 | 2 | No | 432 |
| **class** | 520 | 2 | Positive | 320 |

# Visualisasi data

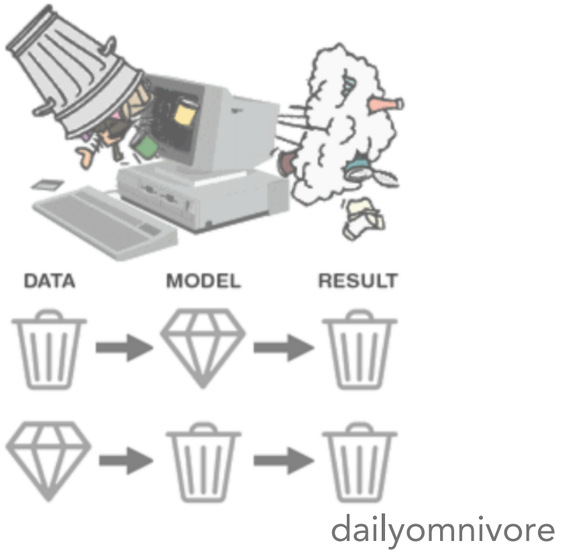
```
import matplotlib.pyplot as plt
df.hist(bins=10, figsize=(10,5))
plt.show()
```





# PERSIAPAN DATA (DATA PREPARATION)

"Garbage in Garbage out"




dailyomnivore

## Pemilihan Data

- Menentukan dataset relevan

## Integrasi & Konstruksi Data

- Mengabungkan data dari berbagai sumber
- Konstruksi data baru

## Pembersihan Data

Melakukan koreksi pada:

- Data duplikat
- Data hilang
- Data keliru, typo
- Konsistensi Format

# PERSIAPAN DATA (DATA PREPARATION)

## Data cleaning
## Eliminasi data duplikat

```
# duplicates check
df.duplicated().sum()
```

```
269
```

```
# drop them
df = df.drop_duplicates()
```

```
# Sanity Check
df.duplicated().sum()
```

```
0
```

## Data Preprocessing (data kategorik --> data numerik)

```
# Encoding # Column Transformer
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
transformer = ColumnTransformer([
    ('onehot', OneHotEncoder(drop='first'), ['Gender', 'Polyuria', 'Polydipsia', 'sudden weight loss',
    'weakness', 'Polyphagia', 'Genital thrush', 'visual blurring', 'Itching', 'Irritability',
    'delayed healing', 'partial paresis','muscle stiffness', 'Alopecia', 'Obesity']),
], remainder='passthrough')
```

## Perubahan format data

```
# Changing format
y = y.replace({'Positive':1, 'Negative': 0})
y.unique()
```

```
array([1, 0], dtype=int64)
```

# PEMBUATAN MODEL MACHINE LEARNING


Beberapa model Machine Learning

### Pemilihan Teknik Modelling

Mencari teknik modeling yg tepat untuk prediksi

### Perancangan Pelatihan model

Merancang pelatihan model dan menentukan komposisi training & tes dataset

### Pembangunan model

Membuat beberapa model & membandingkan performanya

## Tentukan target & fitur, set proporsi pembagian data latihan dan test

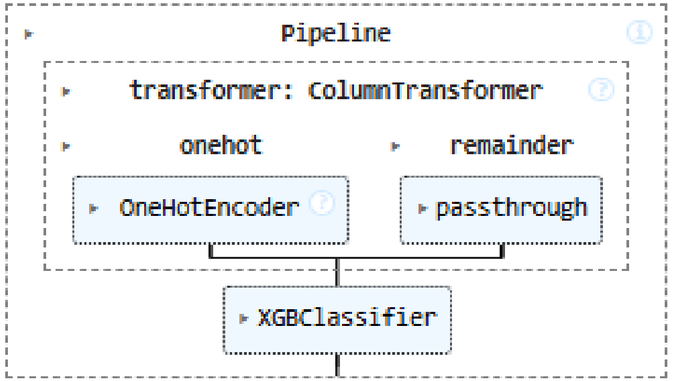
```python
# Define features and target
X = data.drop(columns ='class')
y = data['class']

# Train test split # 30% data test and 70% data train
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0, stratify=y)
```

## Membangun model XGBoost & Random Forest

```python
# Define model XGBoost
from xgboost import XGBClassifier
xgboost = XGBClassifier()
```

```python
# Creating pipeline
from imblearn.pipeline import Pipeline
classification_pipeline_xgb = Pipeline(
    [('transformer', transformer),
     ('model', xgboost)
])
```

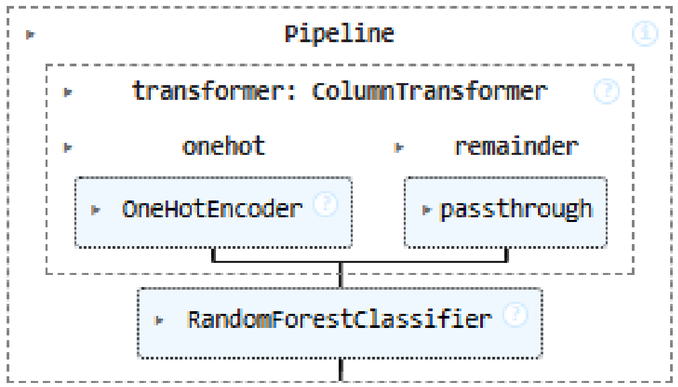
```python
# Fitting model to XGBoost
classification_pipeline_xgb.fit(X_train, y_train)
```




```python
# Define model Random Forest
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier()
```

```python
# Fitting the model
from imblearn.pipeline import Pipeline
classification_pipeline_rfc = Pipeline(
    [('transformer', transformer),
     ('model', rfc)
])
```

```python
# Fitting model to random forest
classification_pipeline_rfc.fit(X_train, y_train)
```



# Evaluation

## Evaluasi Model

- Bagaimana kualitas model?
- Apakah memenuhi standard & kriteria bisnis?

## Review Proses

- Masih adakah kekurangan?
- Apakah semua proses tereksekusi dengan baik
- Buat summary dari temuan model dan koreksi

## Tentukan langkah lanjutan

- Lanjutkan ke deployment?
- Perbaikan/Iterasi ulang model?

# Evaluasi model

XGBoost

```
# XGB model performance
from sklearn.metrics import recall_score

# performance on training data
y_train_pred = classification_pipeline_xgb.predict(X_train)
print(f'Random Forest recall on train data: {recall_score(y_train,y_train_pred):.4f}')

# performance on testing data
y_test_pred = classification_pipeline_xgb.predict(X_test)
print(f'Random Forest recall on test data: {recall_score(y_test,y_test_pred):.4f}')
```

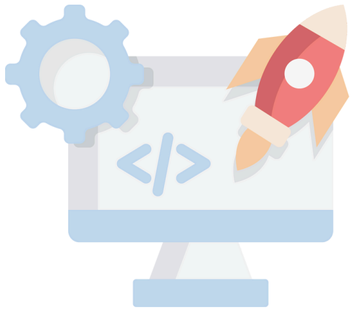
```
Random Forest recall on train data: 0.9917
Random Forest recall on test data: 0.9423
```

Random Forest

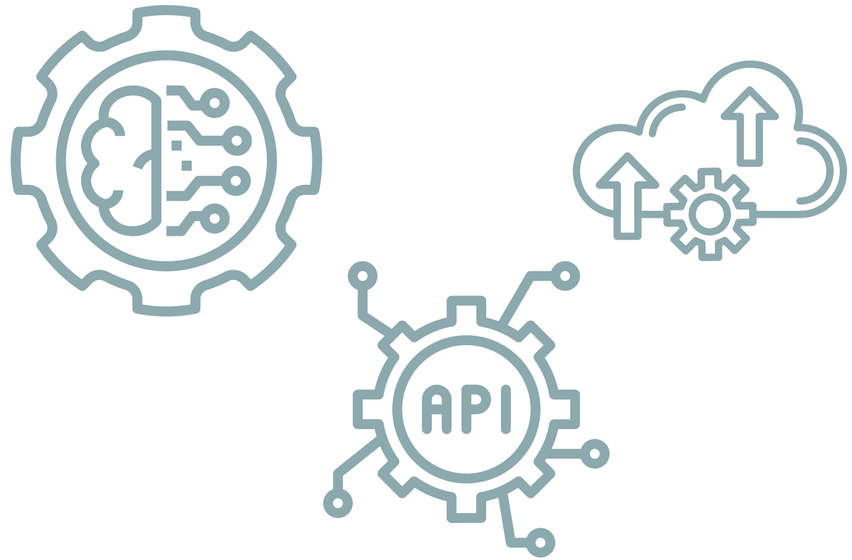
```
# Random forest model performance
from sklearn.metrics import recall_score

# performance on training data
y_train_pred = classification_pipeline_rfc.predict(X_train)
print(f'Random Forest recall on train data: {recall_score(y_train,y_train_pred):.4f}')

# performance on testing data
y_test_pred = classification_pipeline_rfc.predict(X_test)
print(f'Random Forest recall on test data: {recall_score(y_test,y_test_pred):.4f}')
```

```
Random Forest recall on train data: 1.0000
Random Forest recall on test data: 0.9808
```

# MODEL DEPLOYMENT



- Perencanaan deployment
  - Model terpilih disimpan (format pickle)
  - Penggunaan layanan cloud
  - Penyajian API
  - Integrasi ke Apps atau Website
- Monitoring & Maintenance




Website Application

## Menyimpan model

```python
import pickle

# Simpan model
pickle.dump(classification_pipeline_rfc, open("random_forest_diabetes_model.pkl", "wb"))
```

## Membuat API untuk deployment

```python
import pickle
import numpy as np
from flask import Flask, request, jsonify

# Load the trained model
model = pickle.load(open('random_forest_diabetes_model.pkl', 'rb'))

# Initialize Flask app
app = Flask(__name__)

@app.route('/')
def home():
    return "Diabetes Prediction API is running!"

@app.route('/predict', methods=['POST'])
def predict():
    data = request.get_json()  # Expecting JSON input
    features = np.array(data['features']).reshape(1, -1)  # Convert input to numpy array
    prediction = model.predict(features)  # Make prediction
    return jsonify({'prediction': int(prediction[0])})  # Return prediction as JSON

if __name__ == '__main__':
    app.run(debug=True)  # Run API on localhost
```

## Aplikasi pendeteksi risiko diabetes


Prediksi Positif

Prediksi Negatif

# KESIMPULAN

1. CRISP-DM sebagai kerangka alur kerja
2. Pemahaman bisnis sebagai fondasi
3. Pentingnya iterasi dan evaluasi model
4. Deployment sebagai implementasi nyata ML

# THANK YOU

Do you have any question?


My Contact

https://github.com/harishmuh

www.linkedin.com/in/harish-muhammad-7b600b102/

harishmuh@gmail.com