

# SMART RPA powered by Machine Learning

**Charan Kumar** representing Ephicacy, is a technology enthusiast with domain experience of around 15 years.



## Agenda

The history

The mystery

The ...



# The history!!

Adam Smith – Division of Labour

Walter – Bell Labs – Statistical Process Control

Deming and Juran – Quality masterheads

1913

1940s

1990s

1776

1920s

1950s

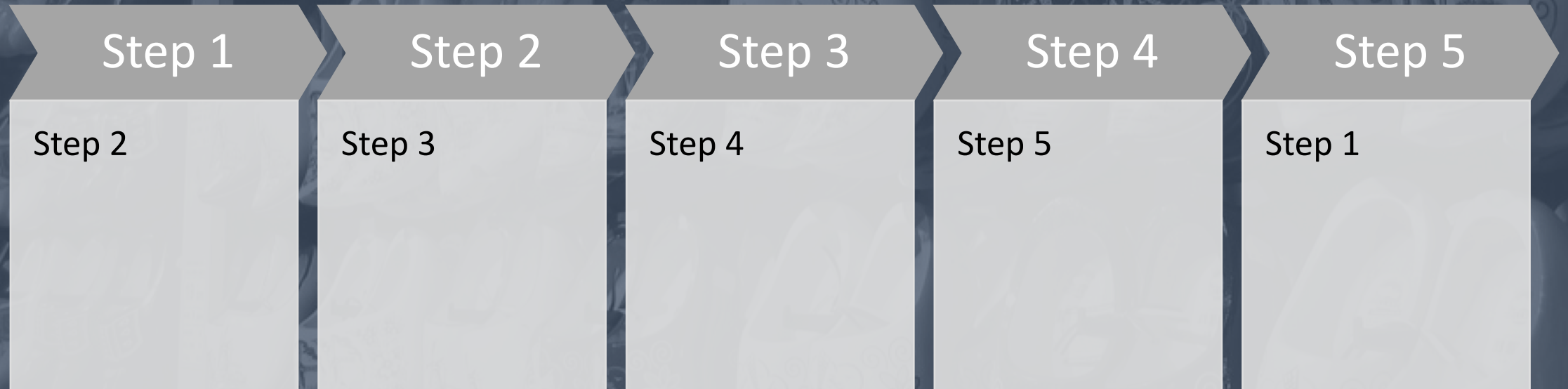
Henry Ford – Assembly belts

Digital Computing

Process digitization – Six Sigma



# The mystery!!





## **The blistery!!**

Data in silos

Change management

Deployment issues

Interoperability among tools/platforms

Manual approach



## **Automation**

BPA

RPA

IPA

**Clinical Development:**

- Pharmacovigilance case processing
- Managing trial master files

**Sales force:**

- Data entry tasks
- Reminders to prospects and customers
- Aggregate past contact history and contact information

**Precision marketing:**

- Providing real-time tailored knowledge to customers
- Virtual medical companion to patients



"""

**Open Excel and reads the searchterms it has to search for.**  
**Opens Chrome and surfs to Google.com and enters searchterm.**  
**Saves all the urls from the first page.**  
**Writes the urls to excel.**

"""

```
from automagica import *

excel_path = "Enter Path to Excel Here" #example: C:\\Users\\Charan\\Desktop\\RPA Test\\data.xlsx

# Read information from the excel in the second row, for columns 2 to 10
lookup_terms = []
for col in range(2,10):
    try:
        print(col)
        lookup_terms.append(ExcelReadCell(excel_path, 2, col))
    except:
        pass

# Open Chrome
browser = ChromeBrowser()

for j,item in enumerate(lookup_terms):
```

```
# Browse to Google
browser.get('https://google.com')
# Lookup the searchterm
browser.find_element_by_xpath('//*[@id="lst-ib"]').send_keys(item)
# Search
browser.find_element_by_xpath('//*[@id="lst-ib"]').submit()
# Get all found items
articles = browser.find_elements_by_class_name("g")
# Parse the headertexts to find the urls
urls = []
for article in articles:
    try:
        import re
        urls.append(re.findall('https?:/(?:[-\w.]|(?:%[\da-fA-F]{2}))+', article.text)[0])
    except:
        pass

# Write found urls to Excel
for i,url in enumerate(urls):
    ExcelWriteCell(excel_path, row=i+2, col=j+2, write_value=url)

# Exit the browser
browser.quit()

# Open the Excel to show result
StartFile(excel_path)
```

```
from automagica import *

# Put the disease names of the first column in a list.
names = ExcelPutColumnInList("C:\\Users\\Charan\\Downloads\\disease.xlsx", "A2","A45")

# Create for every disease a folder with his name and save a .txt file in that folder with his extra information.
for name in names:
    CreateFolder("C:\\Users\\Charan\\Documents\\Subject\\" + ID)
    row = names.index(name) + 2
    data = ExcelPutRowInList("C:\\Users\\Charan\\Downloads\\disease.xlsx", "B"+str(row), "H" + str(row))
    WriteListToFile(data, "C:\\Users\\Charan\\Documents\\disease\\" + name + "\\" + name + ".txt")
```

- **SDTM bots**
- **ADaM bots**



## **Enter the ML!!**

- Decision trees
- Classification models
- Optical character recognition
- Image feature extraction
- Natural language processing

```
# load dataset
pima = pd.read_csv("pima-indians-diabetes.csv", header=None, names=col_names)

#split dataset in features and target variable
feature_cols = ['pregnant', 'insulin', 'bmi', 'age', 'glucose', 'bp', 'pedigree']
X = pima[feature_cols] # Features
y = pima.label # Target variable

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1) # 70% training and 30% test

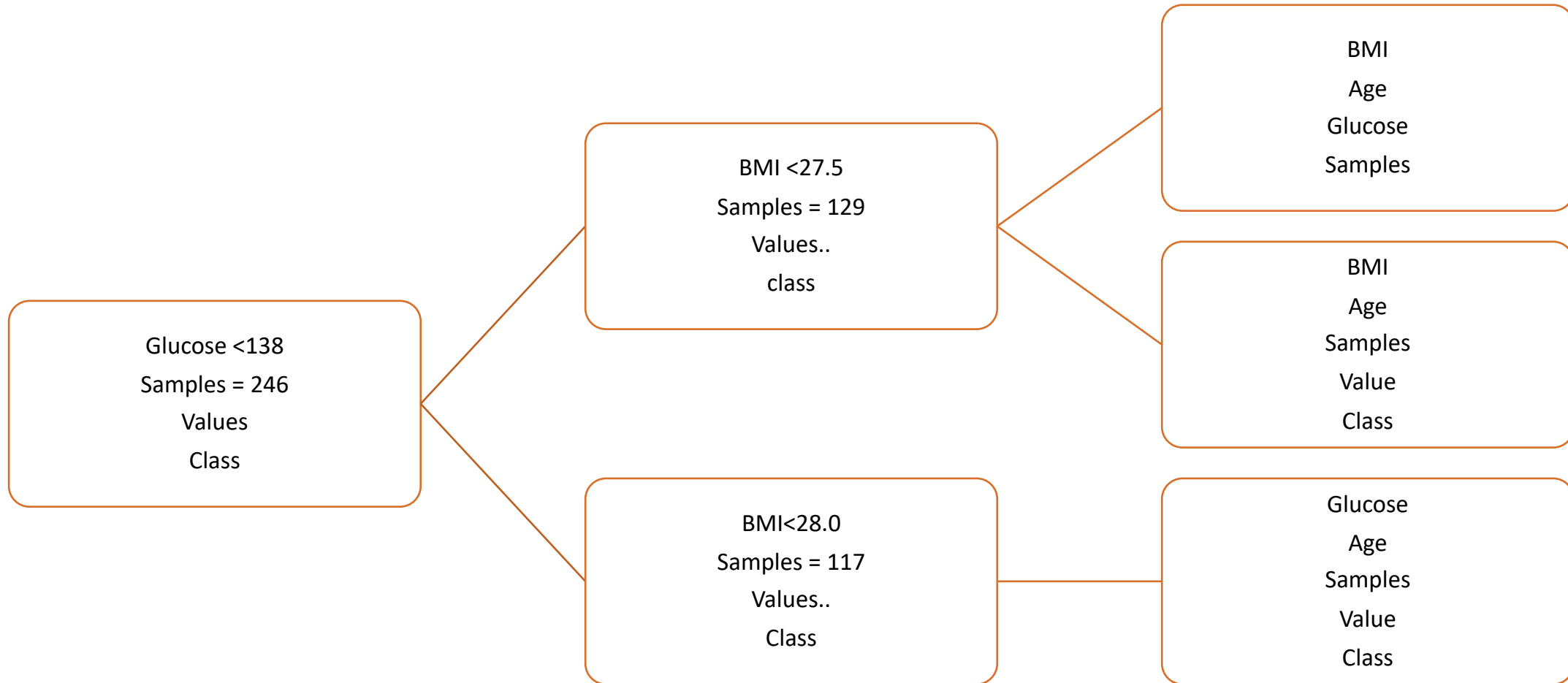
# Create Decision Tree classifier object
clf = DecisionTreeClassifier()

# Train Decision Tree Classifier
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

from sklearn.tree import export_graphviz
from sklearn.externals.six import StringIO
from IPython.display import Image
import pydotplus

dot_data = StringIO()
export_graphviz(clf, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True,feature_names = feature_cols,class_names=['0','1'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('diabetes.png')
Image(graph.create_png())
```



**Applications:**

- Applications in data management
- Applications in pharmacovigilance
- Application in statistical programming
- Application in EHR/EMR
- Clinical Operations



### **Closing thoughts..**

- Transformational efficiency
- Financial yield
- Shorten the journey of drug to market
- Processes - foundations for automation
- Regulatory challenges

Thank you!!

Charan Kumar

[charan.kumar@ephicacy.com](mailto:charan.kumar@ephicacy.com)

[www.ephicacy.com](http://www.ephicacy.com)