Data Science Workshop

Safouene Kallel

2023-12-15

Quizz

SQL is a ... language.

... clause is used to choose the columns

... clause is used to filter the rows

... gets rid of the duplicates

... is used to create aggregated summaries

A Primary key is ...

Quizz

Please complete the Query:

```
SELECT Sales.*, Customers.*
FROM Sales
LEFT JOIN Customers
... Customers.CustomerID = Sales.CustomerID
```

Quizz

{...} helps us connect to SQL
{...} helps us handle data
{...} helps us visualize data

```
pulsedIV <- pulsedIV ...
  filter(Example == 1)
```

Quizz

```
ggplot(pulsedIV,aes(x=V2,
                    y=I,
                    ... = V1)) +
  geom_point() +
  geom_line()
```

Quizz

```
pulsedIV_summary <- pulsedIV %>%
   ...(V1) %>%
   ...(maxI=max(I, na.rm = TRUE))
```

Quizz

What is the difference between numpy and pandas?

Which libraries did we use to connect to SQL?

Which libraries did we use to make a plot?

Please finish the command

```
df1952 = ...
```

Quizz

What Package did we use for building a Web Application?

Please complete the code to create the visual we made last friday.

```
ggplot(PCM_example,
       aes(x=X_COORD,
           y=Y_COORD,
           fill= value)) +
...
```

Quizz

How to make a vector/list/series in R?

```
WaferList <- ...(1,2)
```

Quizz

How to separate between the wafermaps based on the Wafer

```
WaferList <- c(1,2)
PCM_example <- PCM %>%
  filter(Wafer %in% WaferList)

ggplot(PCM_example, aes(x=X_COORD,
                        y=Y_COORD,
                        ...,
                        fill= value)) +
  geom_tile() +
  scale_fill_gradient(low="red",high="green") +
  ...
```

Quizz

What are the two main blocks of a Web Application?
What does each block contain?
How do we transform this to make it work in an interactive code

```
PCM_example <- PCM %>%
    filter(Wafer %in% WaferList)
```

Quizz

Please finish the code:

```
server <- function(input, output) {
  PCM_example <- reactive({PCM %>%
      filter(Wafer %in% WaferList)
  })

  ...plot <- ...({ggplot(..., aes(x=X_COORD,
                                  y=Y_COORD,
                                  group=Wafer,
                                  fill= value)) +
    geom_tile() +
    scale_fill_gradient(low="red",high="green") +
    facet_grid(.~Wafer)
  })
}
```

Summary

We have seen SQL (a neat way to store data but not only).

We have seen 2 open source programming languages:

- We have seen R, packages, SQL connection, pipe, ggplot
- We have seen Python, libraries, SQL connection, seaborn
- We have seen how to transform a static code into an interactive one

We add another building block today.

Summary

- SQL : 1974 - IBM
- R : 1993 (S 1976 Bell Labs)
- RStudio IDE : 2011
- Shiny : 2012
- Tidyverse : 2016
- Python : 1991 The Netherlands
- Numpy : 2006
- Pandas : 2008
- TensorFlow : 2015 - Google

Neural Networks

What we are going to learn today is:

How to create a Neural Network model from numerical data.

Models classification

Theoretical Models:
derived from general Physics laws like Maxwell equations. They cover a wide range of situations and predict extremely well over a very wide range of factor levels.

Empirical Models:
When you can't use a theoretical model, you can create your own empirical model.

General Structure of a Neural Network Script

0. Discuss the requirements with the data owner.
1. Get the data.
2. Normalize it.
3. Split it into Inputs and outputs.
4. Split it into training, validation and testing datasets.
5. Instantiate a Neural Network with a structure.
6. Train the Neural Network.
7. Tune the hyper-parameters to improve learning.
8. Test the model on the testing dataset.
9. Check the accuracy of the model on the testing data.
10. If satisfied with the accuracy save the model.

Neural Network

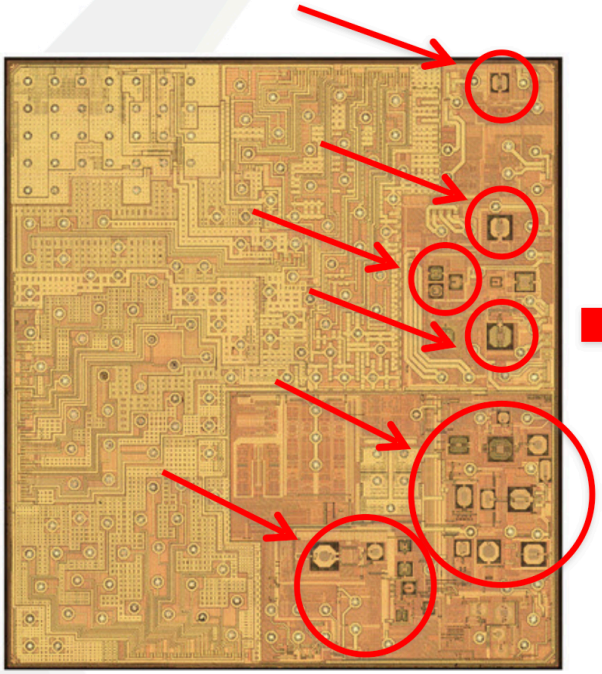
We will use Python and Jupyter to run a Neural Network learning algorithm.

Please open anaconda to start.

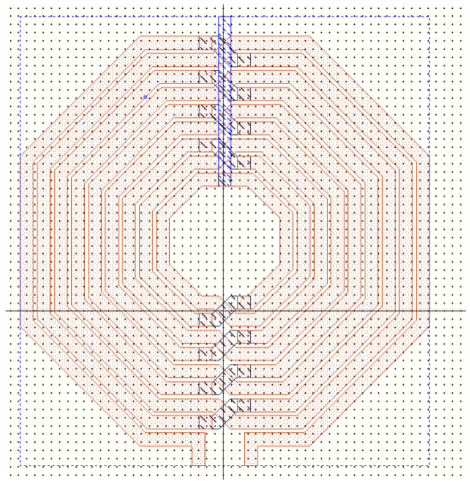
Step 0

Discuss the requirements with the data owner.

Project
Objective



IPhone 5 RF Processor



Geometrical Parameters:
-Diameter
-Number of turns
-Width
-Spacing

Electrical Parameters:
-Inductance
-Quality Factor (Q)
-Self-Resonance Frequency
➔ Electrical Performances

Step 1

importing the usual suspects

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
import math
import keras
```

laoding data

```
inductors = pd.read_csv("./Data/Inductors.csv", sep=";")
```

Step 1

exploring data

```
inductors.describe()


inductors = inductors[["OD","w","n","L"]]
```

Step 2

normalizing data

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

scaler.fit(inductors)

data_scaled = scaler.transform(inductors)
```

Step 3

splitting the data into inputs and outputs

```
X = data_scaled[:,0:3]

Y = data_scaled[:,3]
```

Step 4

splitting the data into training, validation and testing datasets

```
from sklearn.model_selection import train_test_split

X_train, X_val_and_test,
Y_train, Y_val_and_test =
train_test_split(X,Y, test_size=0.3, random_state=0)

X_val, X_test, Y_val, Y_test =
train_test_split(X_val_and_test,
Y_val_and_test, test_size=0.5, random_state=0)
```

Step 5

instantiating a Neural Network with a structure.

```
from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Dropout

model = keras.Sequential([
    keras.Input(shape=(3,)),
    keras.layers.Dense(10, activation="sigmoid"),
    keras.layers.Dense(1, activation="sigmoid")
])
```

Step 6

train the Neural Network

```
model.compile(optimizer = 'adam',
              loss = 'mean_squared_error',
              metrics = ['MeanSquaredError']
)

hist = model.fit(X_train,Y_train,
                 batch_size=64,
                 epochs = 10000,
                 validation_data = (X_val,Y_val))
```

Step 7

testing the model on the testing dataset.

```
from tensorflow.keras.metrics import MeanSquaredError

model.evaluate(X_test,Y_test)[1]

plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.yscale('log')
plt.xlabel('Epoch')
plt.legend(['Train','Val'], loc='upper right')
plt.show
```

Step 8

tuning the hyper-parameters to improve learning

```
model = keras.Sequential([
    keras.Input(shape=(3,)),
    keras.layers.Dense(10, activation="tanh"),
    keras.layers.Dense(1, activation="tanh")
])
```

Step 9

```
y_pred = model.predict(X)

plot = pd.DataFrame({'Actual' : Y.flatten(), 'Predicted' : y_p

plot.plot(x = 'Actual',
y = 'Predicted',
style = 'o',
color = 'blue')
plt.xscale('log')
plt.yscale('log')
plt.show()
```

Step 10

saving the model

```
model.save('InductorsML.h5')
```

Homework

Create a Quarto pdf or html document where you describe your dataset and what you want to extract from it.
Mix prose, code and output.
Visualize your summary in a plot.
Generate a pdf that you can send to people.
Work in groups.
Share your work in the DS Community Team Channel.
If you find difficulties setting it up, please send me an email. I can discuss this with you later.

Deadline end of October.

Thank you