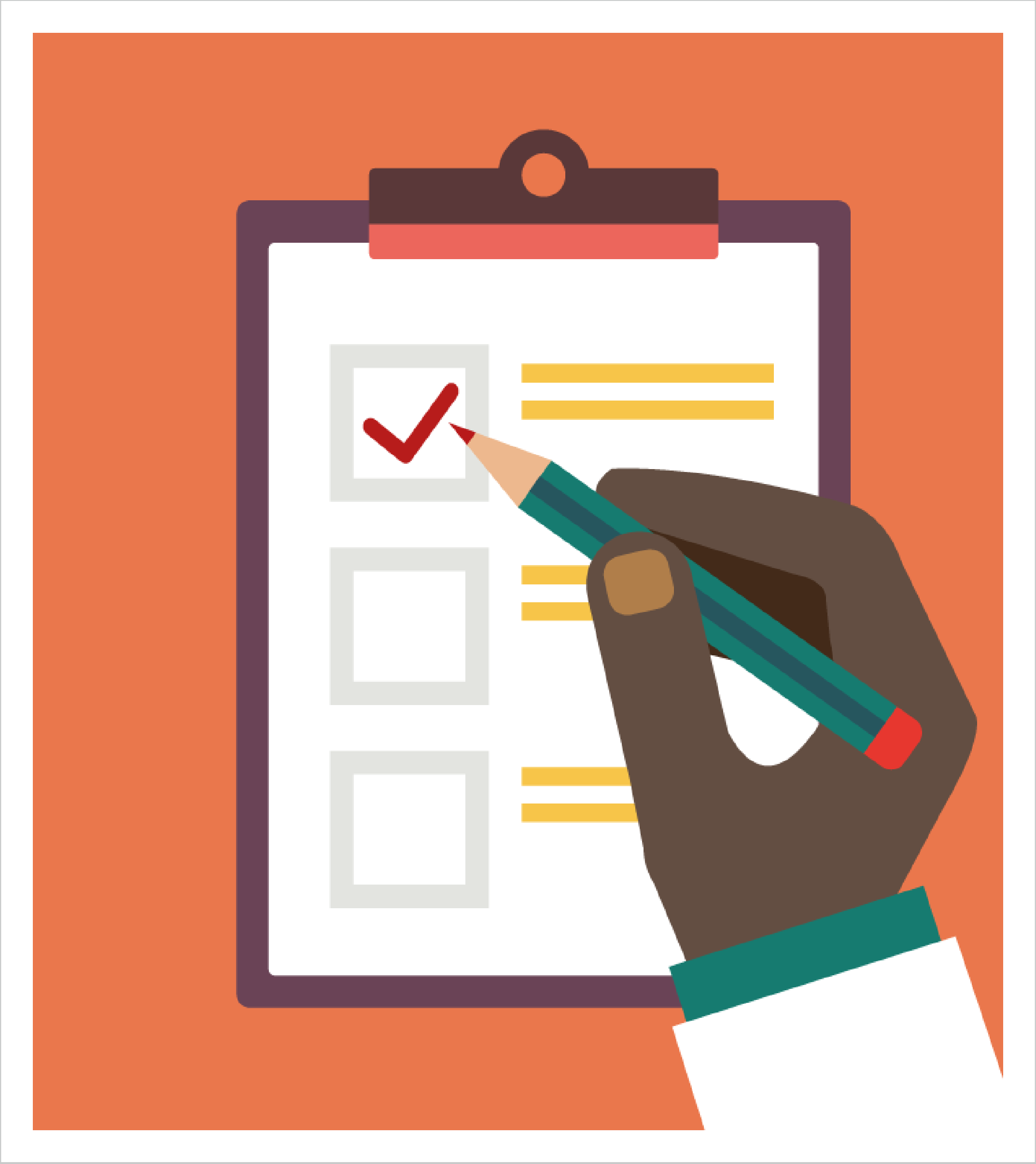
RAPIDS

Accelerating ML using RAPIDS

INSAID



Agenda

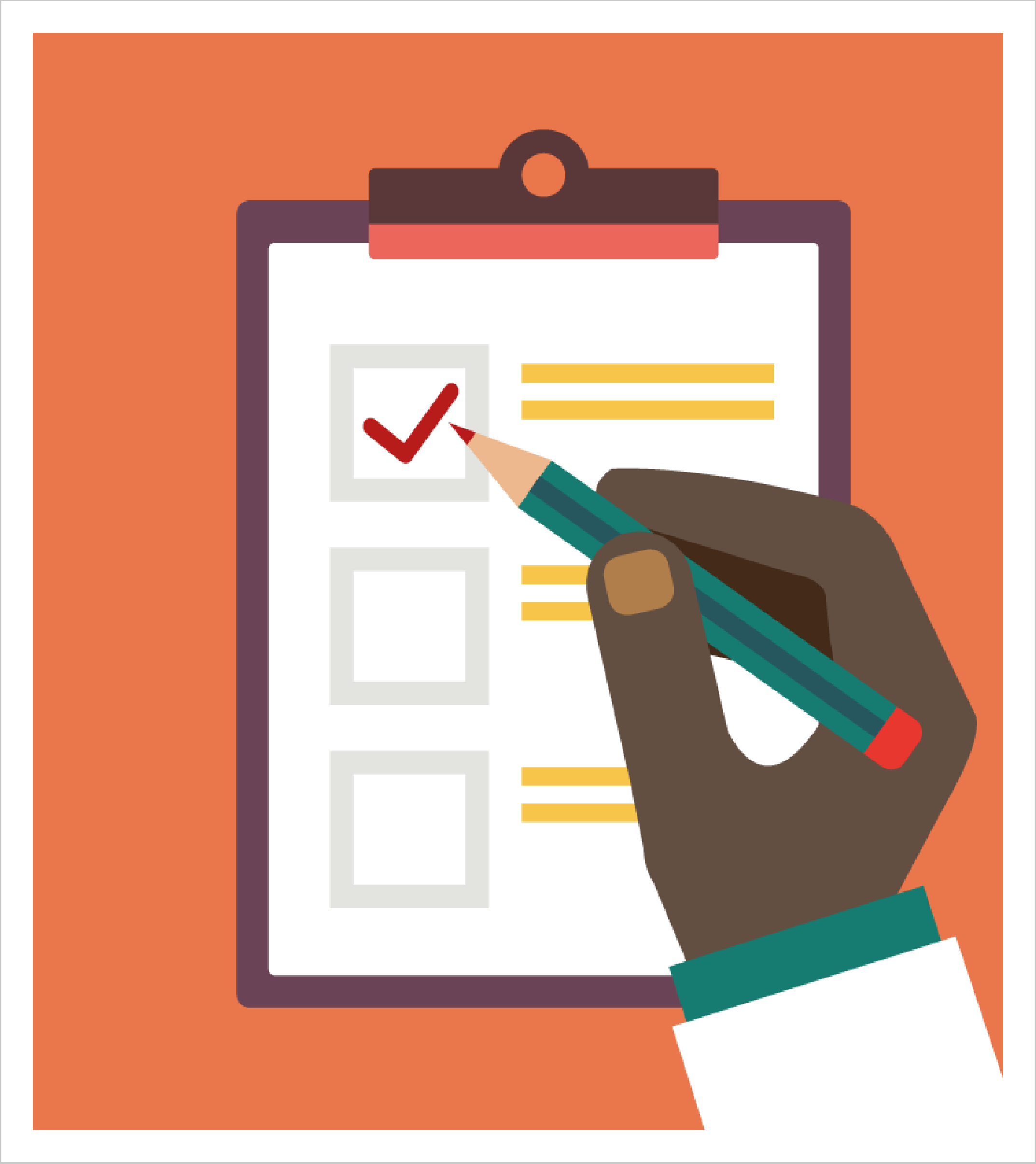
- **Why Rapids?**
- Why GPU?
- Libraries in RAPIDS
- ML Pipeline using RAPIDS
- Pandas vs cuDF
- Scikit-learn vs cuML
- Demo case study

INSAID

Why RAPIDS?

- **Rapids** is a suite of libraries designed for accelerating Data Science

- The beauty of Rapids is that it’s integrated smoothly with Data Science libraries —things like Pandas data-frames are easily passed through to Rapids for GPU acceleration.

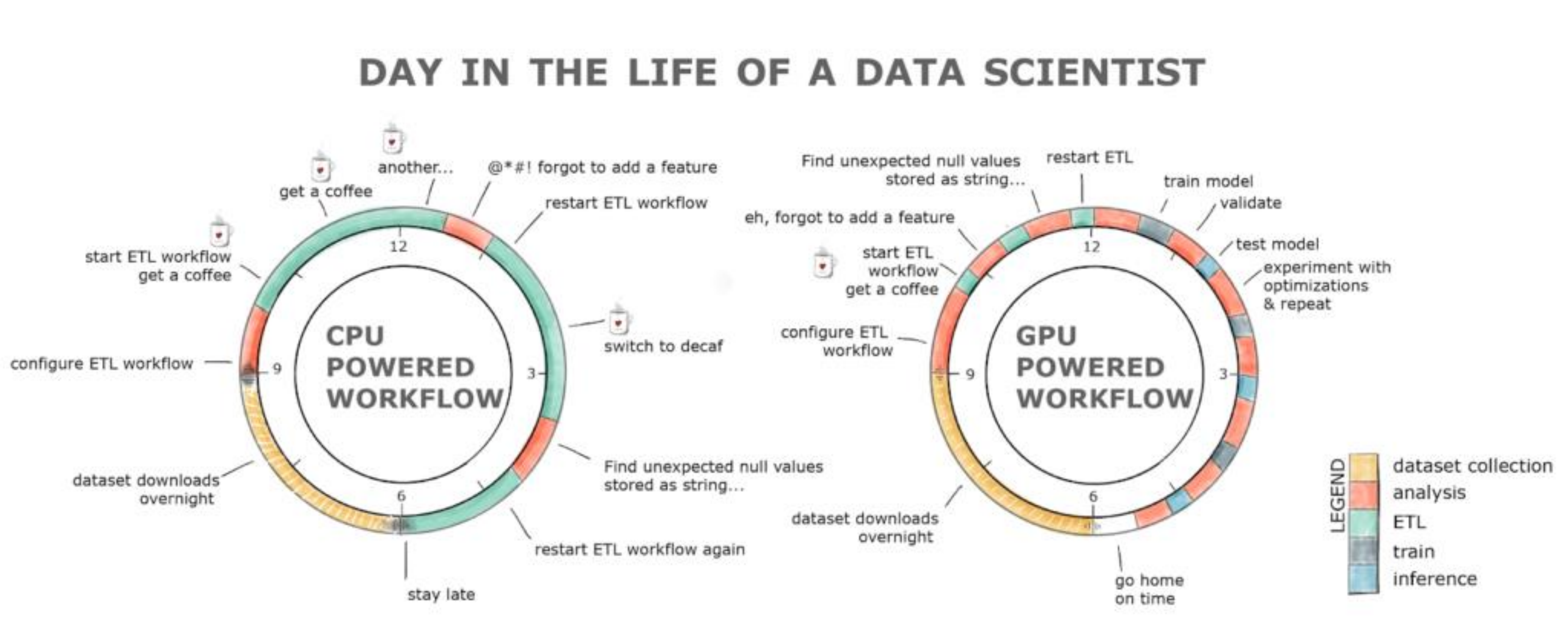


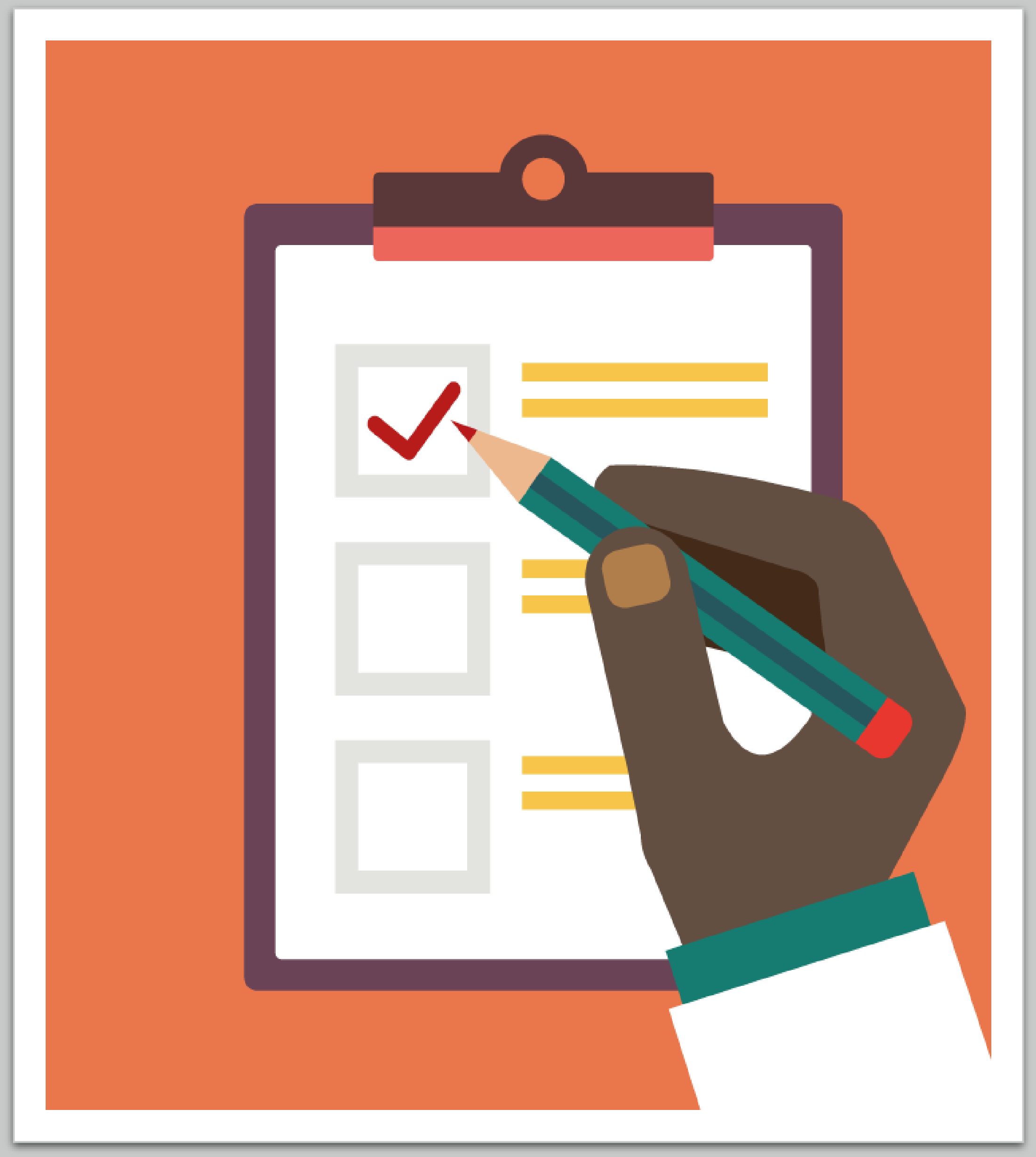
Agenda

- Why Rapids?
- **Why GPU?**
- Libraries in RAPIDS
- ML Pipeline using RAPIDS
- Pandas vs cuDF
- Scikit-learn vs cuML
- Demo case study

INSAID

Why GPU?


DAY IN THE LIFE OF A DATA SCIENTIST
CPU POWERED WORKFLOW
get a coffee
another...
@*#! forgot to add a feature
restart ETL workflow
start ETL workflow
get a coffee
12
switch to decaf
configure ETL workflow
9
3
6
Find unexpected null values
stored as string...
dataset downloads
overnight
restart ETL workflow again
stay late
GPU POWERED WORKFLOW
Find unexpected null values
stored as string...
restart ETL
train model
validate
eh, forgot to add a feature
test model
start ETL
workflow
get a coffee
experiment with
optimizations
& repeat
configure ETL
workflow
dataset downloads
overnight
go home
on time
LEGEND
dataset collection
analysis
ETL
train
inference

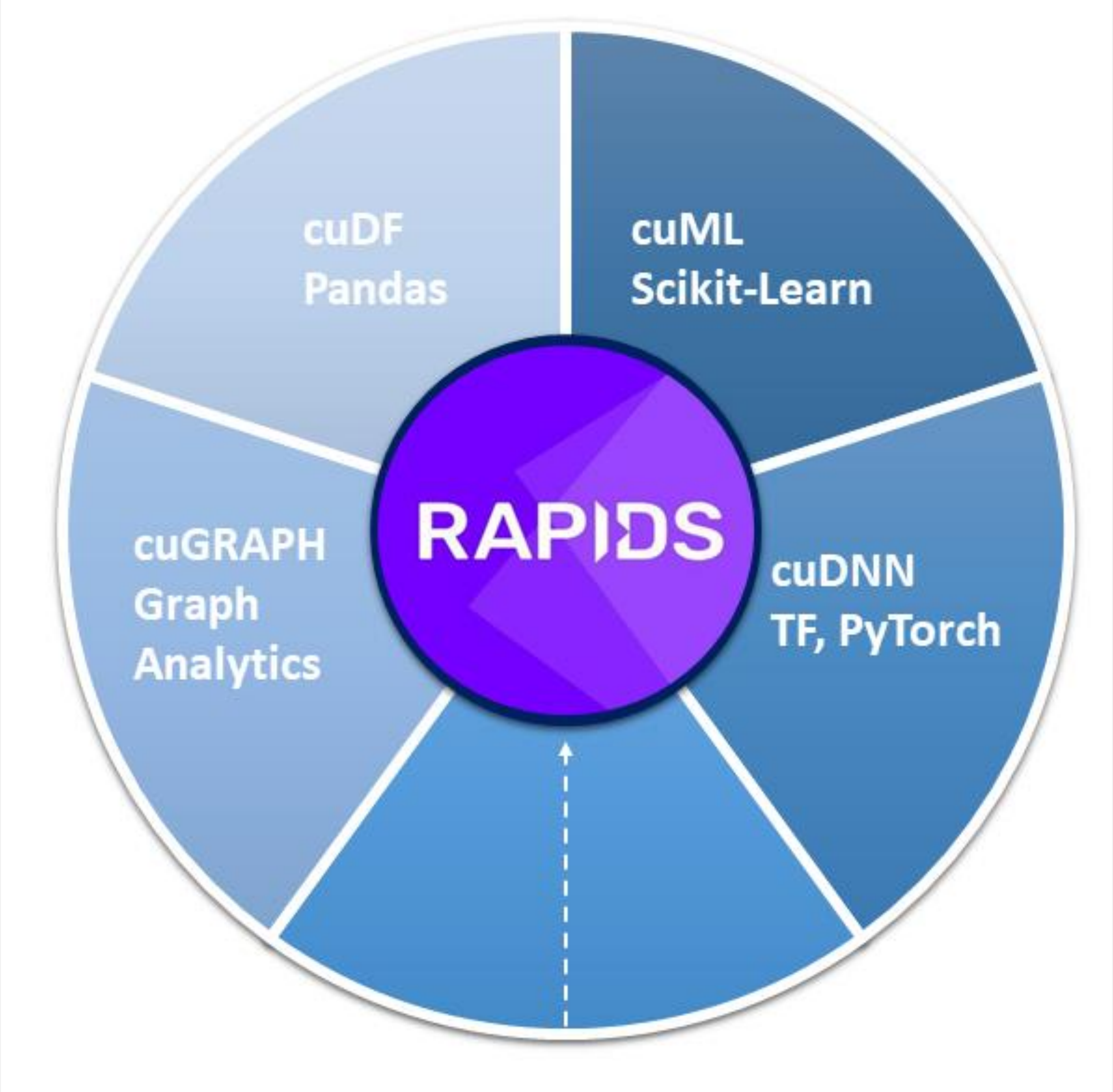


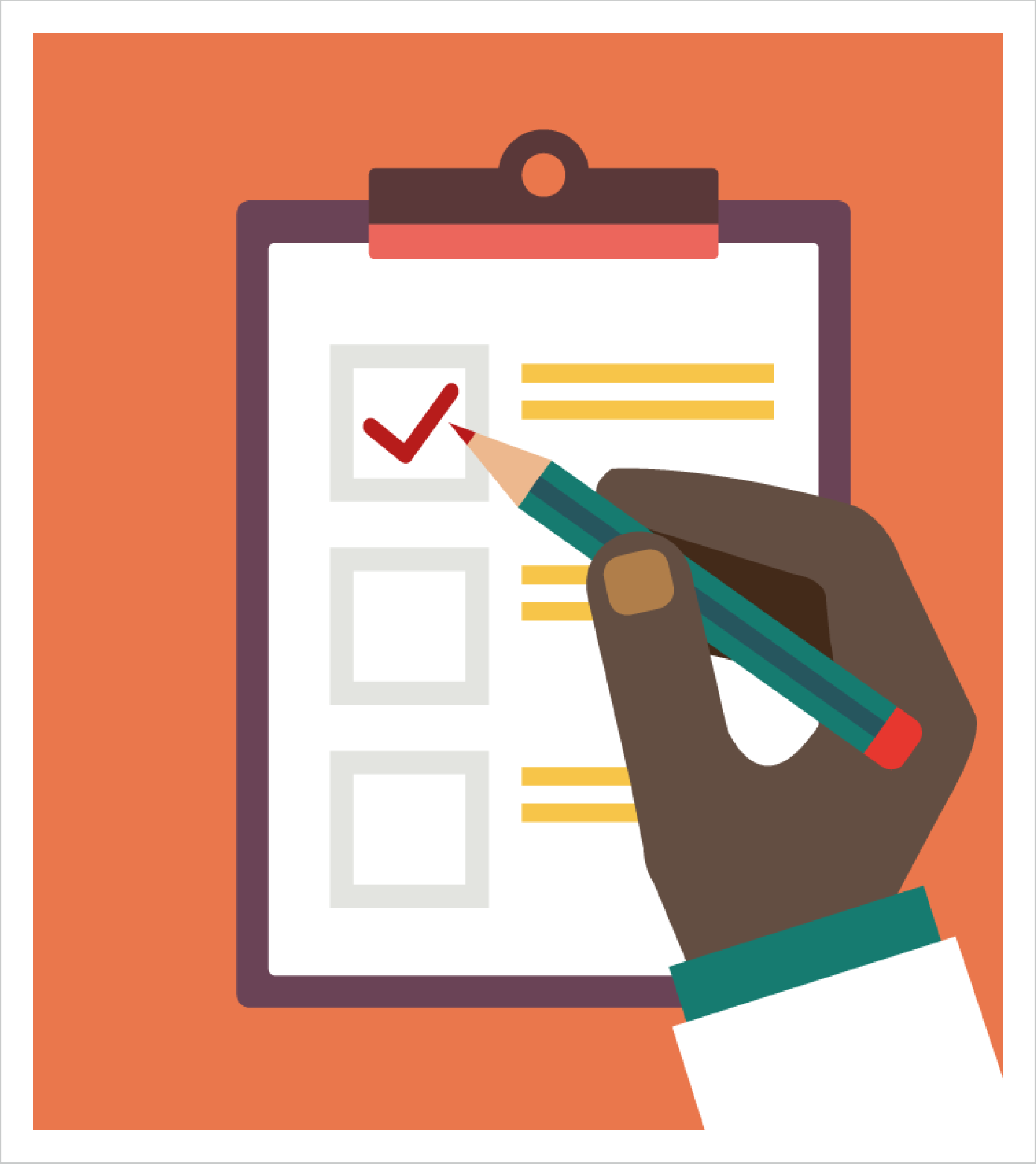
Agenda

- Why Rapids?
- Why GPU?
- **Libraries in RAPIDS**
- ML Pipeline using RAPIDS
- Pandas vs. CuDF
- Scikit-learn vs. CuML
- Demo case study

INSAID

Libraries in RAPIDS

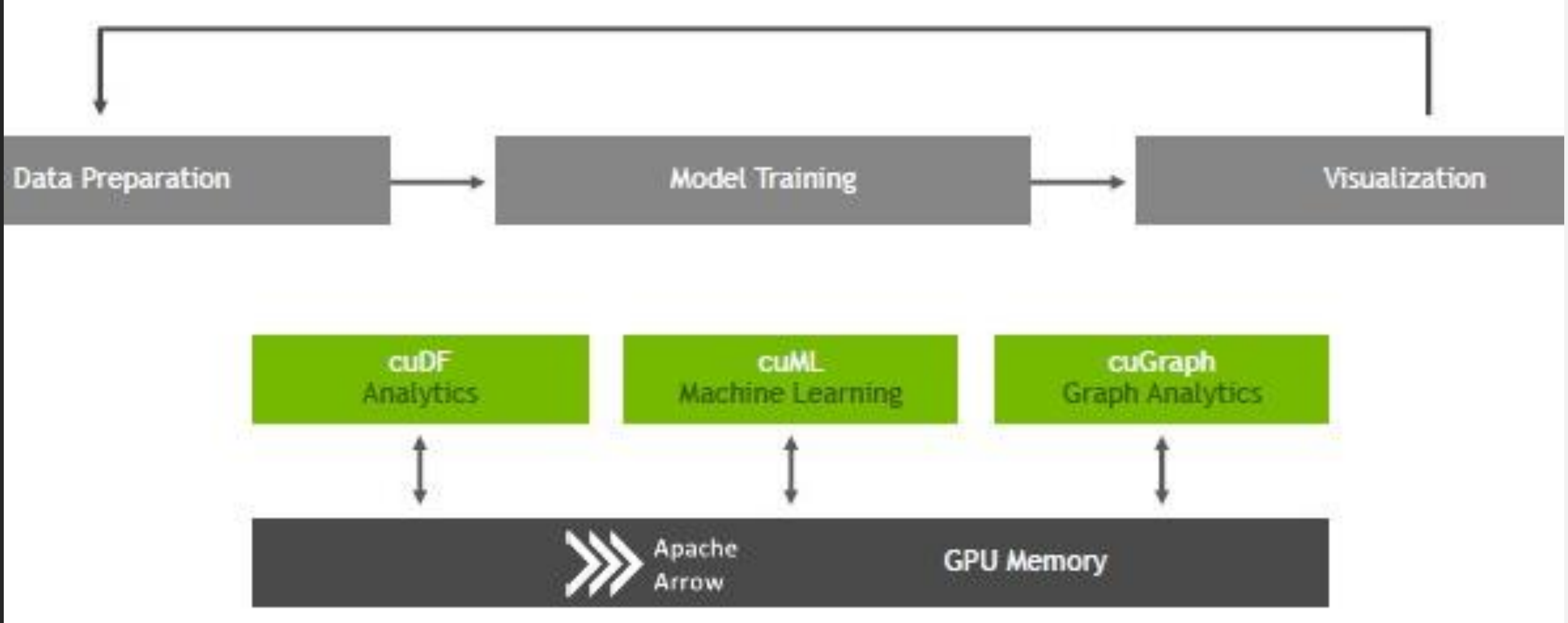


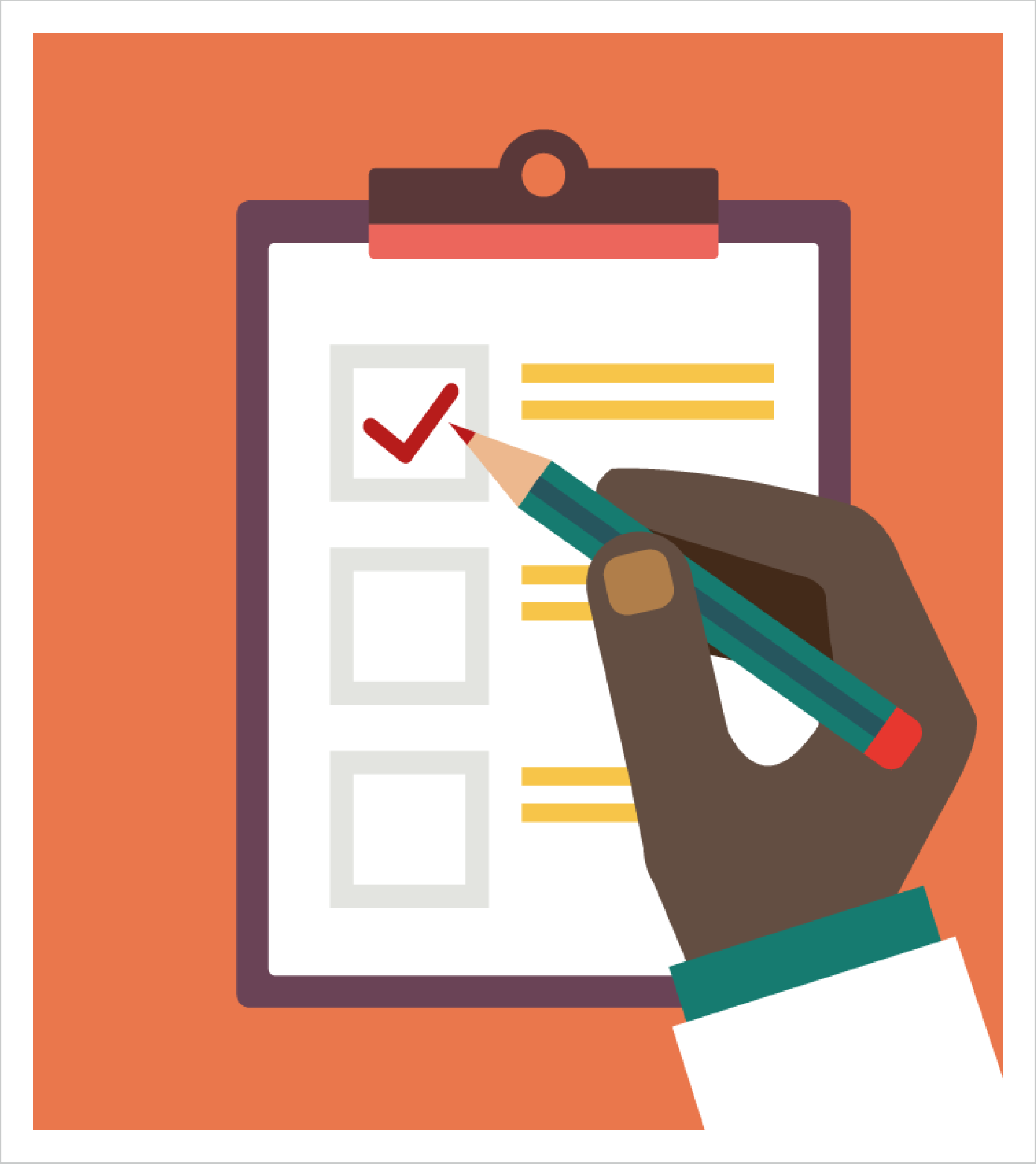


Agenda

- Why Rapids?
- Why GPU?
- Libraries in RAPIDS
- **ML Pipeline using RAPIDS**
- Pandas vs cuDF
- Scikit-learn vs cuML
- Demo case study

ML Pipeline using RAPIDS





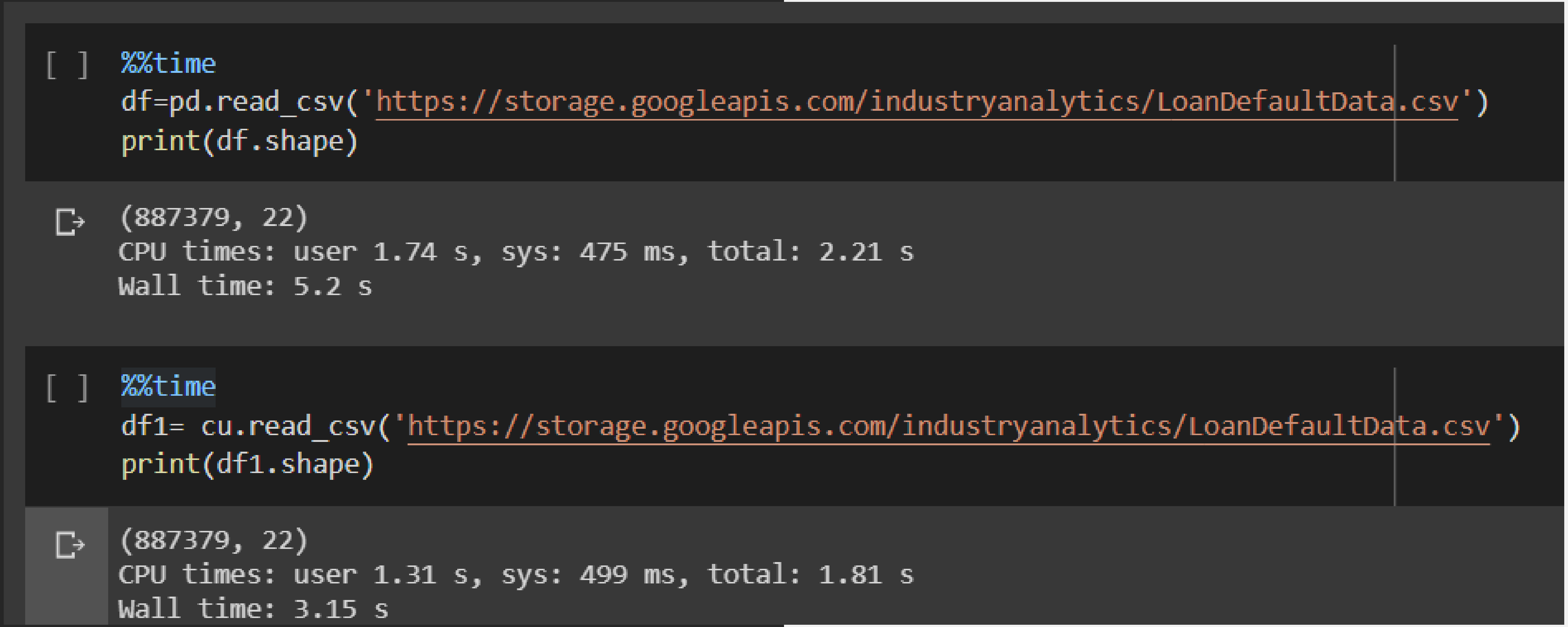
Agenda

- Why Rapids?
- Why GPU?
- Libraries in RAPIDS
- ML Pipeline using RAPIDS
- **Pandas vs cuDF**
- Scikit-learn vs cuML
- Demo case study

INSAID

Pandas vs cuDF

- cuDF always outperforms Pandas while importing a big dataset

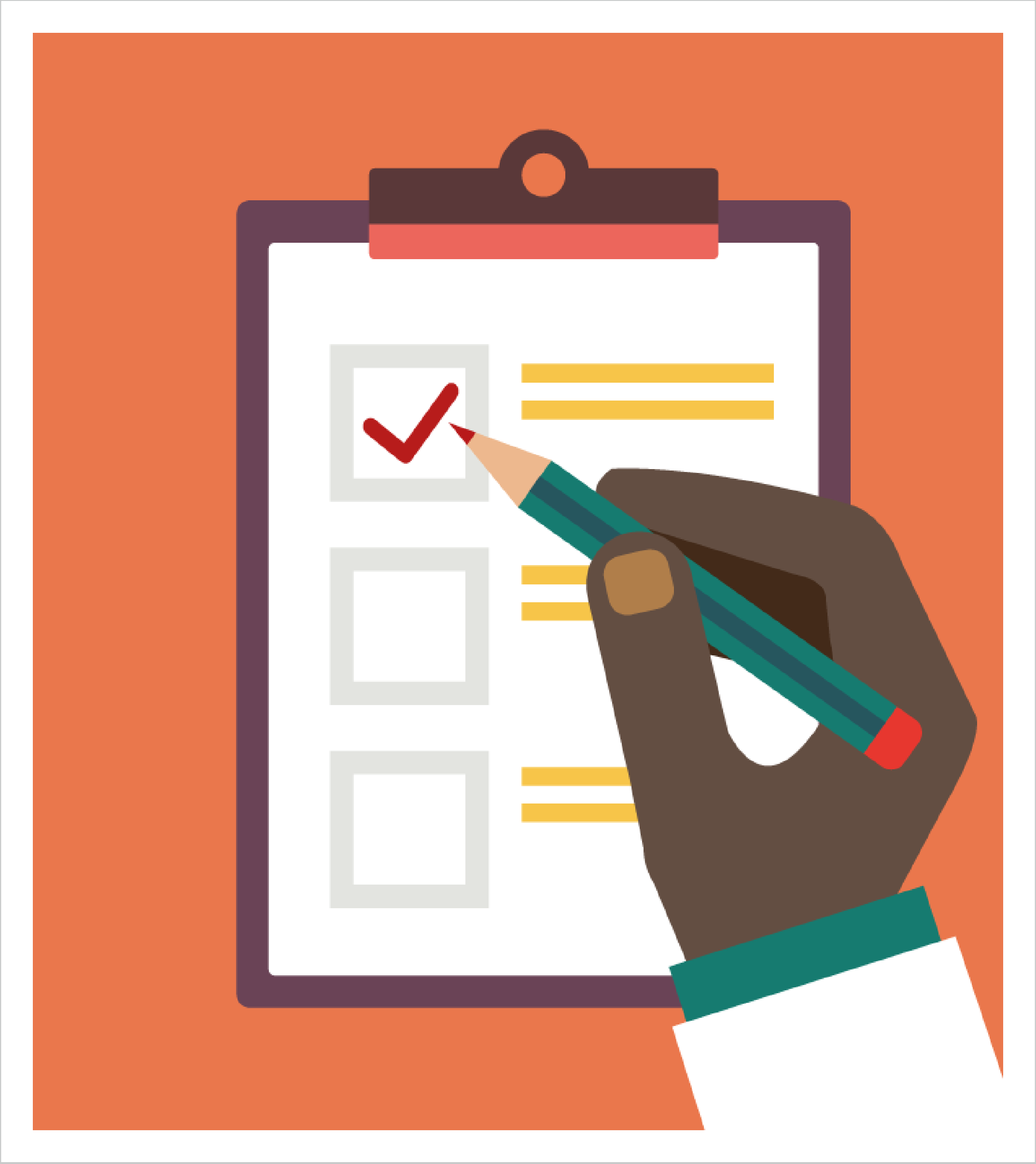


```
%%time
df=pd.read_csv('https://storage.googleapis.com/industryanalytics/LoanDefaultData.csv')
print(df.shape)
```

```
(887379, 22)
CPU times: user 1.74 s, sys: 475 ms, total: 2.21 s
Wall time: 5.2 s
```

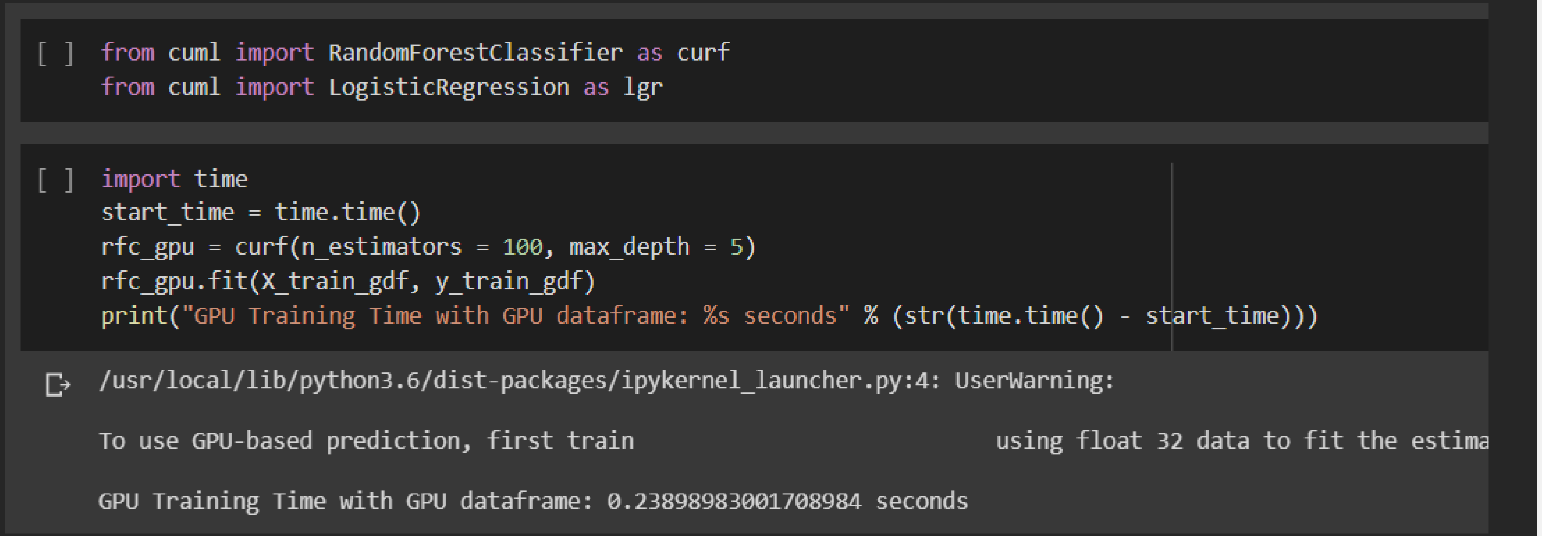
```
%%time
df1= cu.read_csv('https://storage.googleapis.com/industryanalytics/LoanDefaultData.csv')
print(df1.shape)
```

```
(887379, 22)
CPU times: user 1.31 s, sys: 499 ms, total: 1.81 s
Wall time: 3.15 s
```



Agenda

- Why Rapids?
- Why GPU?
- Libraries in RAPIDS
- ML Pipeline using RAPIDS
- Pandas vs cuDF
- **Scikit-learn vs cuML**
- Demo case study



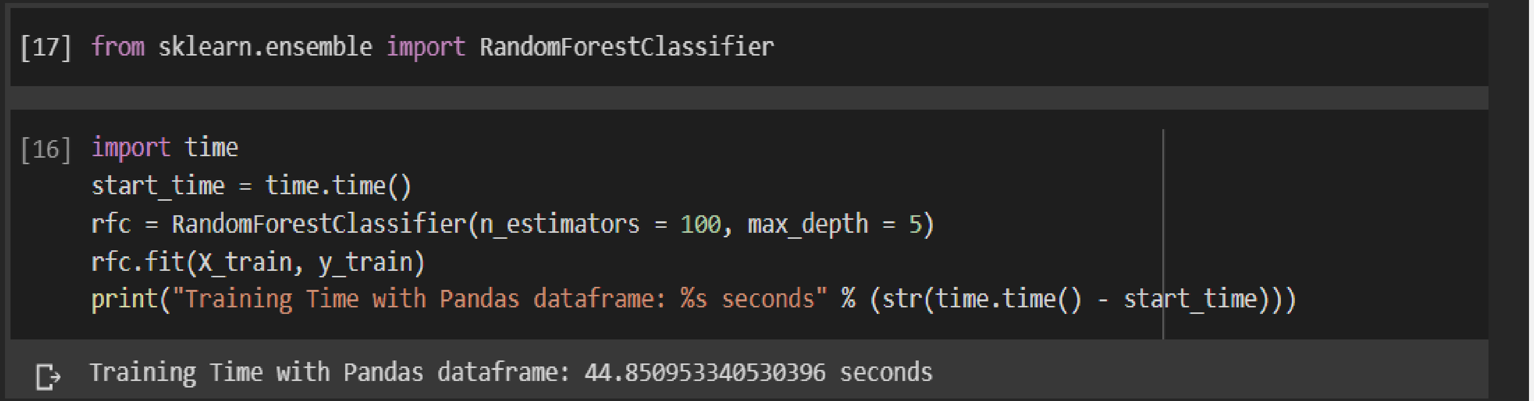
```
[ ] from cuml import RandomForestClassifier as curf
    from cuml import LogisticRegression as lgr
```

```
[ ] import time
    start_time = time.time()
    rfc_gpu = curf(n_estimators = 100, max_depth = 5)
    rfc_gpu.fit(X_train_gdf, y_train_gdf)
    print("GPU Training Time with GPU dataframe: %s seconds" % (str(time.time() - start_time)))
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: UserWarning:

To use GPU-based prediction, first train                          using float 32 data to fit the estima

GPU Training Time with GPU dataframe: 0.23898983001708984 seconds
```



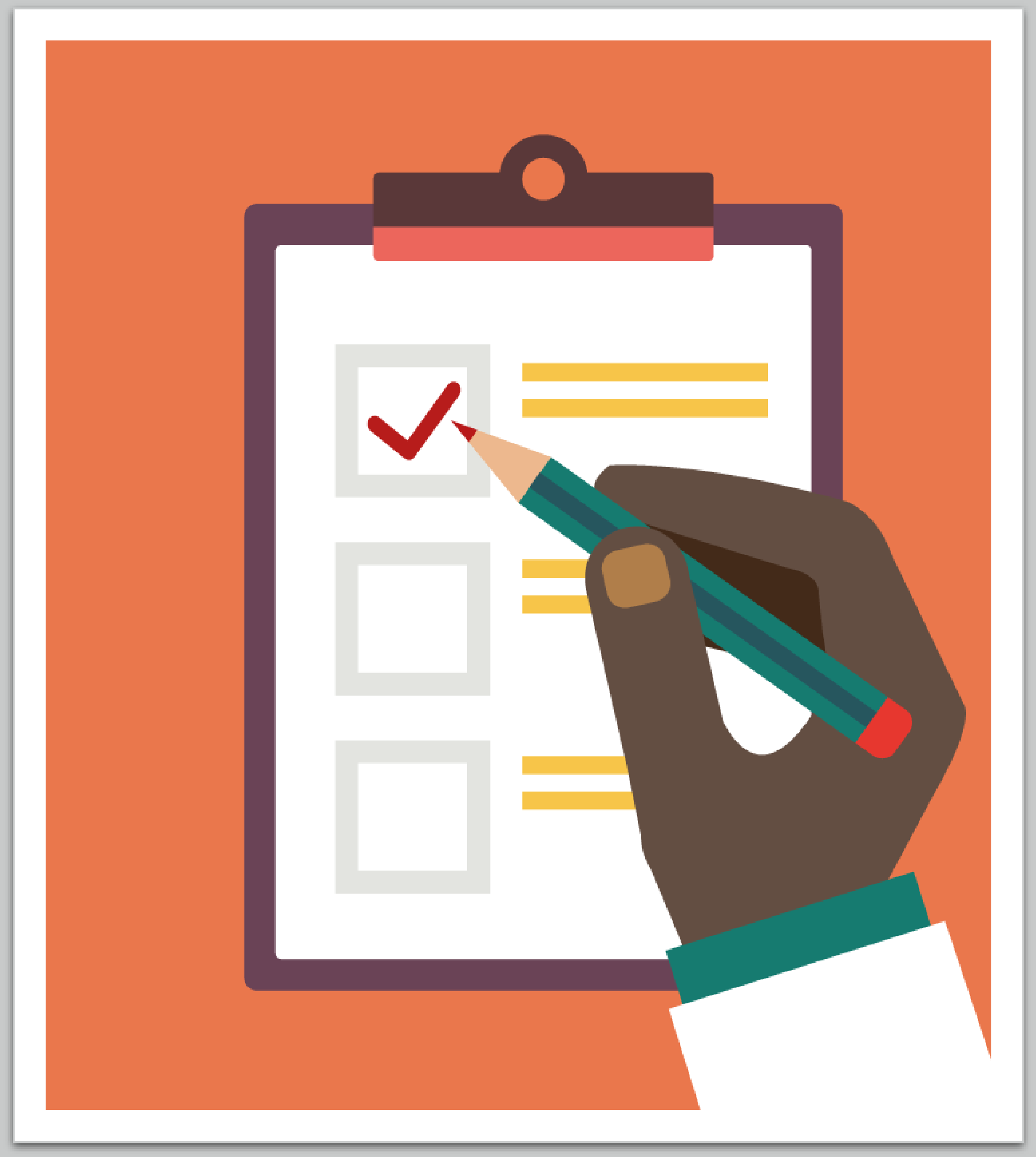
```
[17] from sklearn.ensemble import RandomForestClassifier
```

```
[16] import time
     start_time = time.time()
     rfc = RandomForestClassifier(n_estimators = 100, max_depth = 5)
     rfc.fit(X_train, y_train)
     print("Training Time with Pandas dataframe: %s seconds" % (str(time.time() - start_time)))
```

```
Training Time with Pandas dataframe: 44.850953340530396 seconds
```

Scikit-learn vs. cuML

- Random-Forest algorithm from cuML outperforms the traditional Scikit-learn Random-Forest algorithm with a speed 44 times higher than the traditional Random forest algorithm



Agenda

- Why Rapids?
- Why GPU?
- Libraries in RAPIDS
- ML Pipeline using RAPIDS
- Pandas vs cuDF
- Scikit-learn vs cuML
- **Demo case study**

THANK
YOU