Vectorization (向量化)

- A super fast alternative to loops in Python.

Example

```
import numpy as np
import pandas as pd
df = pd.DataFrame(np.random.randint(0, 100, size=(10000000, 4)),
                  columns=('English','Math','Physics','Biology'))
df.shape
df.head()
```

```
In [9]: df.head()
Out[9]:
   English  Math  Physics  Biology
0       53    74       71       49
1       71    17       36       72
2       53    16       71       57
3        5    21       27       70
4       30    68       82       20

In [10]: df.shape
Out[10]: (100000000, 4)
```

Example (using loops)

```
import time
start = time.time()


# Iterating using iterrows
for idx, row in df.iterrows():
    # creating a new column
    df.at[idx,'avg'] = (row["English"]+row["Math"]
                        +row["Physics"]+row["Biology"])/4

end = time.time()
print(end - start)
```

```
In [12]: runfile('C:/Users/josep/_Project/
vectorization_example.py', wdir='C:/Users/josep/
_Project')
358.01435947418213

In [13]:
```

Example (using vectorization)

```
import time
start = time.time()
df["ratio"] = df["English"]+df["Math"]
                          +df["Physics"]+df["Biology"])/4


end = time.time()
print(end - start)
```

```
In [17]: runfile('C:/Users/josep/_Project/
vectorization_example.py', wdir='C:/Users/josep/
_Project')
0.0643925666809082

In [18]:
```

Example (using vectorization)
Approach #2

```
import time
start = time.time()
df["ratio"] = df[["English", "Math", "Physics", "Biology"]].mean(axis=1)

end = time.time()
print(end - start)
```

```
In [11]: runfile('C:/Users/josep/_Project/
vectorization_example.py', wdir='C:/Users/josep/
_Project')
0.14576983451843262

In [12]:
```

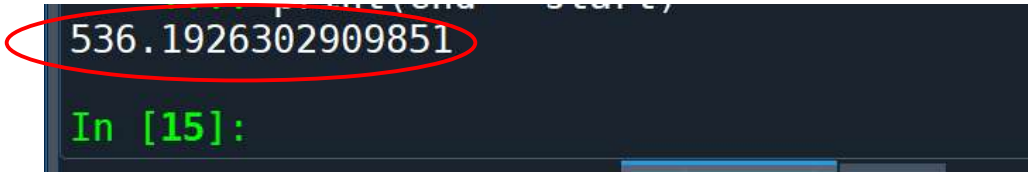
If else condition (using loops)

```
start = time.time()

# Iteration using iterrows
for idx, row in df.iterrows():
    if (row.Math > row.Physics):
        df.at[idx,'final'] = (row.Math + row.English + row.Biology)/3
    else:
        df.at[idx,'final'] = (row.Physics + row.English + row.Biology)/3

end = time.time()

print(end - start)
```



If else condition (Vectorization)

```
start = time.time()

df['final'] = (df['Physics'] + df['English'] + df['Biology'])/3
df.loc[df['Math'] > df['Physics'], 'final'] =
                            (df['Math'] + df['English'] + df['Biology'])/3

end = time.time()

print(end - start)
```

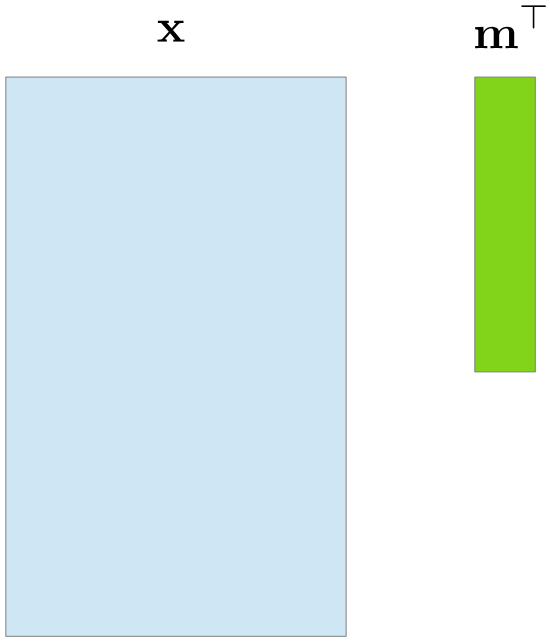
```
In [20]: runfile('C:/Users/josep/_Project/
vectorization_example.py', wdir='C:/Users/josep/_Project')
0.30089688301086426

In [21]:
```

Matrix multiplication

$$
\begin{aligned}
y_i &= \langle \mathbf{m}, \mathbf{x}_i \rangle \\
&= m_1 x_{i,1} + m_2 x_{i,2} + \ldots + m_k x_{i,k}
\end{aligned}
$$

```
m = np.random.rand(1,5)
x = np.random.rand(5000000,5)
#assume k=5
```



Matrix multiplication

```
start = time.time()
zer = []


for i in range(0,5000000):
    total = 0
    for j in range(0,5):
        total = total + x[i][j]*m[0][j]


    zer.append(total)
zer = np.array(zer)


end = time.time()
print ("Computation time = " + str(en
```

```
In [8]: runfile('C:/Users/josep/_Project/
vectorization_matrix.py', wdir='C:/Users/josep/_Project')
Computation time = 13.515385389328003 seconds
```

Matrix multiplication (vectorization)

```
start = time.time()
zer = np.matmul(x, m.T)
end = time.time()
print ("Computation time = " + str(end-start) + " seconds")
```

```
In [13]: runfile('C:/Users/josep/_Project/
vectorization_matrix.py', wdir='C:/Users/josep/_Project')
Computation time = 0.01042532920837402 seconds
```