
Stay Fit
Stay Healthy
FitMe
by Ameya, Akshit, Harshit, Jeremie

Background & Use Case

Having entered the new year recently, there is a lot of buzz about adopting healthy lifestyles recognizing its physical and mental benefits.

Consequently, monitoring results becomes an integral part of one’s lifestyle-changing process in order to successfully accomplish fitness goals.

Our goal is to deploy a tool that helps users stay in tune with their health goals like sleep, activity levels, weight management and caloric requirements using their smart watch data.

Each of these health goals constitute individual use cases, where users will be presented with relevant insights that help them monitor and accomplish their goals.

Why do we need python libraries?

We have multi-feature, tabular data which is cumbersome to process line-by-line using standard python code

- EDA libraries to parse through them
- Visualization libraries to convey insights pictorially
- Machine learning libraries to predict caloric requirements using our rich data sources

Python package choices

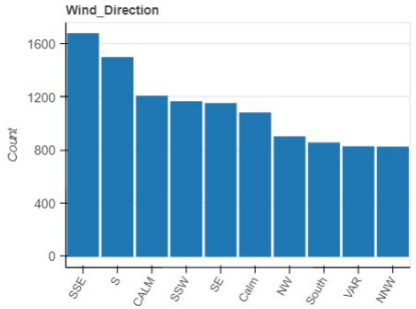
DataPrep by SFU database system lab

Fastest and the easiest EDA tool in Python. It analyzes a Pandas / Dask DataFrame with a few lines of code in seconds.

Wind_Direction
categorical

Show Details

Approximate Distinct Count	24
Approximate Unique (%)	0.2%
Missing	117
Missing (%)	0.7%
Memory Size	1.0 MB

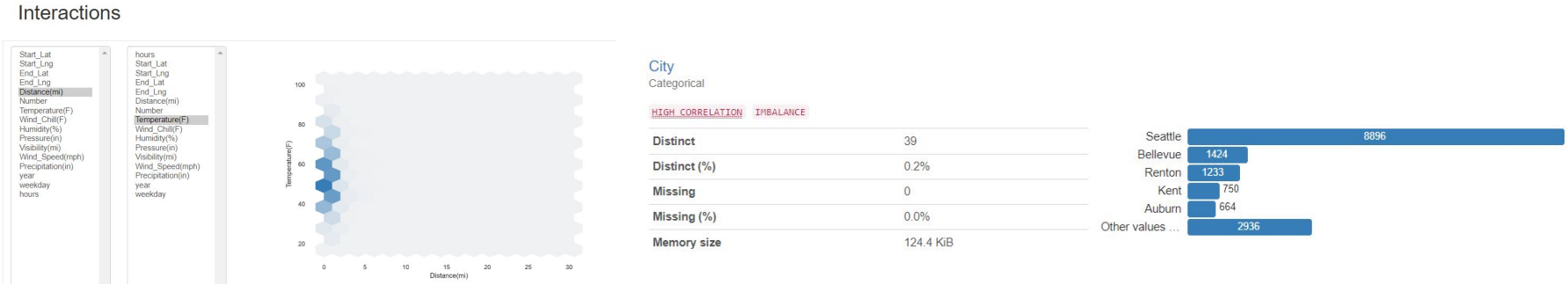


Python package choices

Pandas profiling by YData Labs Inc.

Provides a one-line EDA experience in a consistent manner. Like pandas df.describe() function, pandas-profiling delivers an extended analysis of a DataFrame.

The package outputs a simple and digested analysis of a dataset, including time-series and text.



Python package choices

	DataPrep	**Pandas Profiling**
Use	Suitable for EDA	Suitable for profiling
Speed	100X faster due to parallelization	Slower due to sequential processing
Interactivity	Allows users to easily format and customize the default visualizations to suit their design requirements	Have fixed charts and lack interactivity. For instance, correlations between variables are displayed with heat maps only
Memory	Supports out of core processing	Limited due to within-core processing

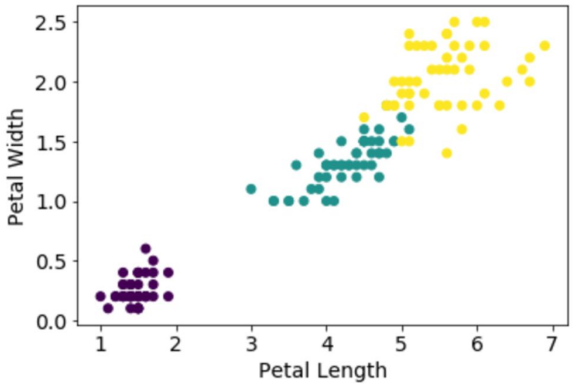
Python package options

Matplotlib	**Seaborn**
Creates simple graphs, including bar graphs, histograms, pie charts, scatter plots, etc	Additionally, seaborn employs engaging themes, and it helps in the integration of all data into a single plot
Matplotlib is a Python graphics package for data visualization and integrates nicely with Numpy and Pandas	Seaborn is more comfortable with Pandas data frames. It utilizes simple sets of techniques to produce lovely images in Python
Matplotlib is highly customized and robust	With the help of its default themes, Seaborn prevents overlapping plots
It utilizes syntax that is relatively complicated and extensive eg: Matplotlib.pyplot.bar(x-axis, y-axis) is the syntax for a bar graph	It has relatively simple syntax, making it simpler to learn and comprehend eg: seaborn.barplot(x axis, y-axis) syntax for a bar graph

Python package choices

Scikit learn by David Cournapeau as a Google Summer of Code project

Python module for machine learning built on top of SciPy. Scikit-learn supports most of the supervised and unsupervised learning algorithms.




```
#Import knearest neighbors Classifier model
from sklearn.neighbors import KNeighborsClassifier

#Create KNN Classifier
knn = KNeighborsClassifier(n_neighbors=5)

#Train the model using the training sets
knn.fit(X_train, y_train)

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

Python package choices

Tensorflow by Google

Train and run very large neural networks efficiently based on deep learning algorithms by distributing the computations across potentially thousands of multi-GPU servers.

Low-level library that is used for deep learning models, unlike scikit-learn which can be considered as the high-level library used to train classical machine learning models.

```
2019-10-26 20:24:36.992185: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU
ports instructions that this TensorFlow binary was not compiled to use: AVX2
Train on 200000 samples
Epoch 1/10
200000/200000 [==============================] - 6s 29us/sample - loss: 5228847.5627
Epoch 2/10
200000/200000 [==============================] - 6s 28us/sample - loss: 10115.1791
Epoch 3/10
200000/200000 [==============================] - 6s 28us/sample - loss: 2250.9930
Epoch 4/10
200000/200000 [==============================] - 6s 28us/sample - loss: 1728.0065
Epoch 5/10
200000/200000 [==============================] - 6s 28us/sample - loss: 1365.0211
Epoch 6/10
200000/200000 [==============================] - 6s 28us/sample - loss: 1117.7696
Epoch 7/10
200000/200000 [==============================] - 6s 28us/sample - loss: 988.0515
```

```
1  model1 = tf.keras.Sequential()
2  model1.add(tf.keras.layers.Dense(64, input_shape=(2,) , activation='sigmoid'))
3  model1.add(tf.keras.layers.Dense(128, activation = 'relu'))
4  model1.add(tf.keras.layers.Dense(2))
5  model1.compile(optimizer=tf.keras.optimizers.Adam(0.001), loss=tf.keras.losses.MSE)
6  model1.fit(x=x, y=y, epochs=EPOCHS)
```

Python package choices

TensorFlow	**Scikit learn**
It is a low-level library for implementing ML techniques and algorithms	A higher-level library for implementing and comparing ML algorithms
TensorFlow is a deep learning framework	Scikit-learn is mostly used in machine learning applications
Mainly used to assist developers designing architectures and benchmarking their models	Used to create and benchmark new models, provides built-in functions for most supervised and unsupervised ML algorithms

Final package choices

With the end product being an analytical tool hosted using Streamlit, we have decided on the following packages as they are compatible, robust and are capable of addressing all our use cases

1. DataPrep
    - Better suited for EDA
    - Customizable
    - Low-latency

2. Seaborn
    - Ease of integration
    - Easier syntax
    - Better suited for Pandas dataframes

3. Scikit-learn
    - Better suited for ML models compared to DL centric TensorFlow
    - Using DL would be an overkill, similar problems like ours have been solved using ML

Concerns

- Integrating everything using Streamlit
  We might have to deploy our tool locally owing to time constraints. We will provide detailed documentation on how to deploy our tool locally

- Lack of domain knowledge
  Choosing the most relevant variables might be arduous

- Large scope
  We might have to limit the scope of this project depending on some challenges we encounter. **Scope is subject to change.**

  We’re planning to create a visualization dashboard as well as a CLI.

Concerns

- Integrating everything using Streamlit
  We might have to deploy our tool locally owing to time constraints. We will provide detailed documentation on how to deploy our tool locally

- Lack of domain knowledge
  Choosing the most relevant variables might be arduous

- Large scope
  We might have to limit the scope of this project depending on some challenges we encounter. **Scope is subject to change.**

  We’re planning to create a visualization dashboard as well as a CLI.

Demo (CLI - useful for local testing & debugging)

Package usage example:

./stayfit.py <command> <metric> <options>

<command>:
- visualize # outputs a graph with time as x-axis for the desired metric
- export # outputs raw data into specified format
- stats # outputs statistics about the desired metric for the time frame

<metric>:
- calories # tracks calories burnt
- heartbeat # track heartbeat sensor
- steps # track steps
- sleep # track sleep
- weight # track weight

<options>:
- –from # lower bound of time range selection
- –to # upper bound of time range selection
- –format: csv | pickle # export format selection
- –daily # shorthand to specify the time range to be [ago(1d), now()]
- –out # file path to export data

Demo

Using seaborn boxplots to see variations between activity minutes and day of week

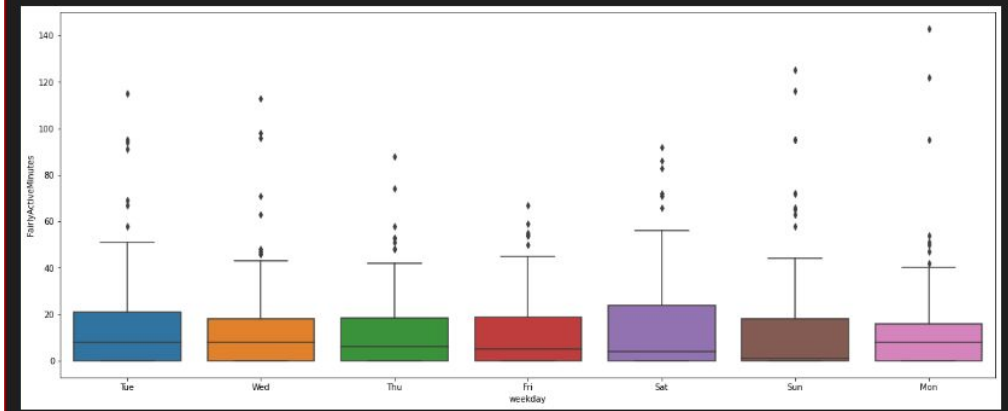
```
df['weekday'] = df['weekday'].map({0: 'Mon', 1: 'Tue', 2: 'Wed', 3: 'Thu', 4: 'Fri', 5: 'Sat', 6: 'Sun' })
```

0.0s Python

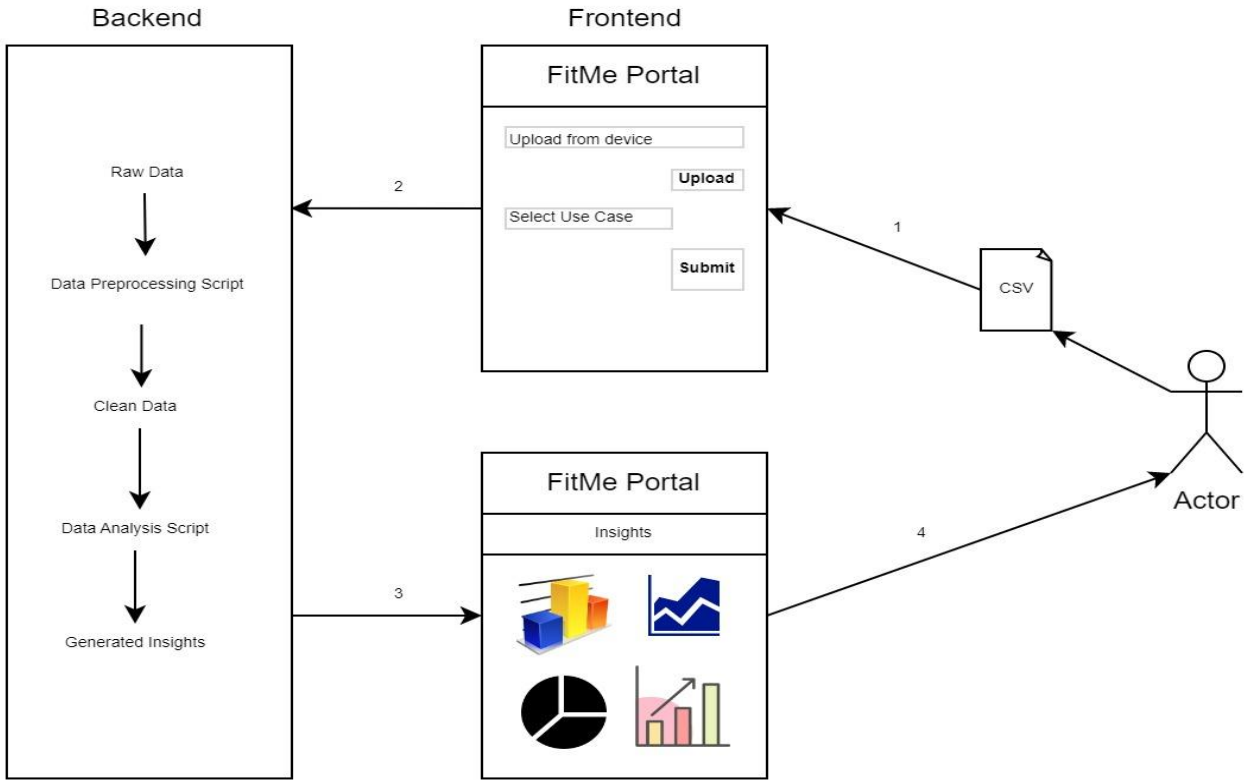
```
plt.figure(figsize=(20, 8))
sns.boxplot(x=df["weekday"], y=df["FairlyActiveMinutes"])
```

0.5s Python

```
<AxesSubplot:xlabel='weekday', ylabel='FairlyActiveMinutes'>
```



Demo


Backend
Frontend
FitMe Portal
Upload from device
Upload
Select Use Case
Submit
Raw Data
Data Preprocessing Script
Clean Data
Data Analysis Script
Generated Insights
2
1
CSV
Actor
FitMe Portal
Insights
3
4

Project deliverables timeline

March 02, 2023	Data preprocessing script
March 04, 2023	Data analysis scripts for all use-cases
March 06, 2023	UI design and development
March 08, 2023	System integration
March 10, 2023	Comprehensive unit testing
March 12, 2023	Final submission