Machine Learning in Python

High Level Steps

- Import the Data
- Clean the Data
- Split the Data into Training/Test Sets (80% training/20% testing)
- Create a Model - select an algorithm
- Train the Model
- Make Predictions
- Evaluate and Improve

Libraries and Tools

Library	Purpose
NumPy	NumPy offers comprehensive mathematical functions, random number generators, linear algebra routines, Fourier transforms, and more.
Pandas	Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool.
MatPlotLib	Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
SciKit-Learn	Simple and efficient tools for predictive data analysis · Accessible to everybody, and reusable in various contexts
Jupyter	The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser.
Anaconda	Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment.

Getting Started

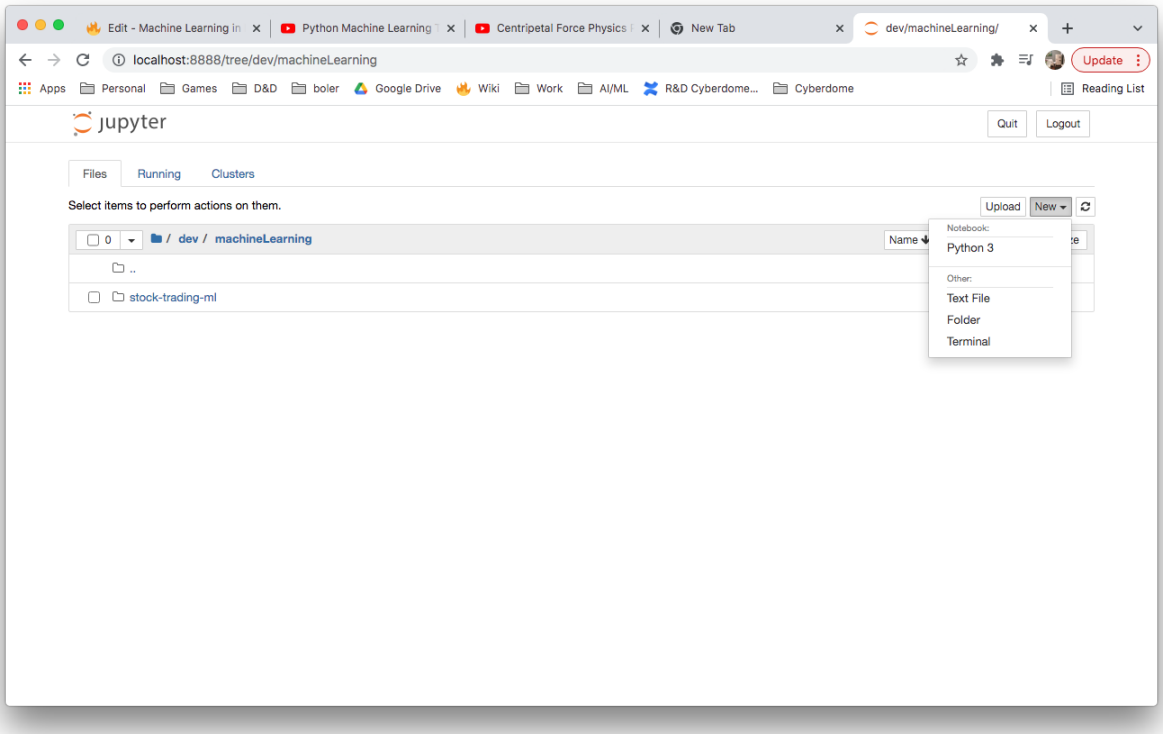
Install Anaconda

https://www.anaconda.com/products/individual

Start a jupyter notebook

```
$ jupyter notebook
```

Create a new Python3 notebook



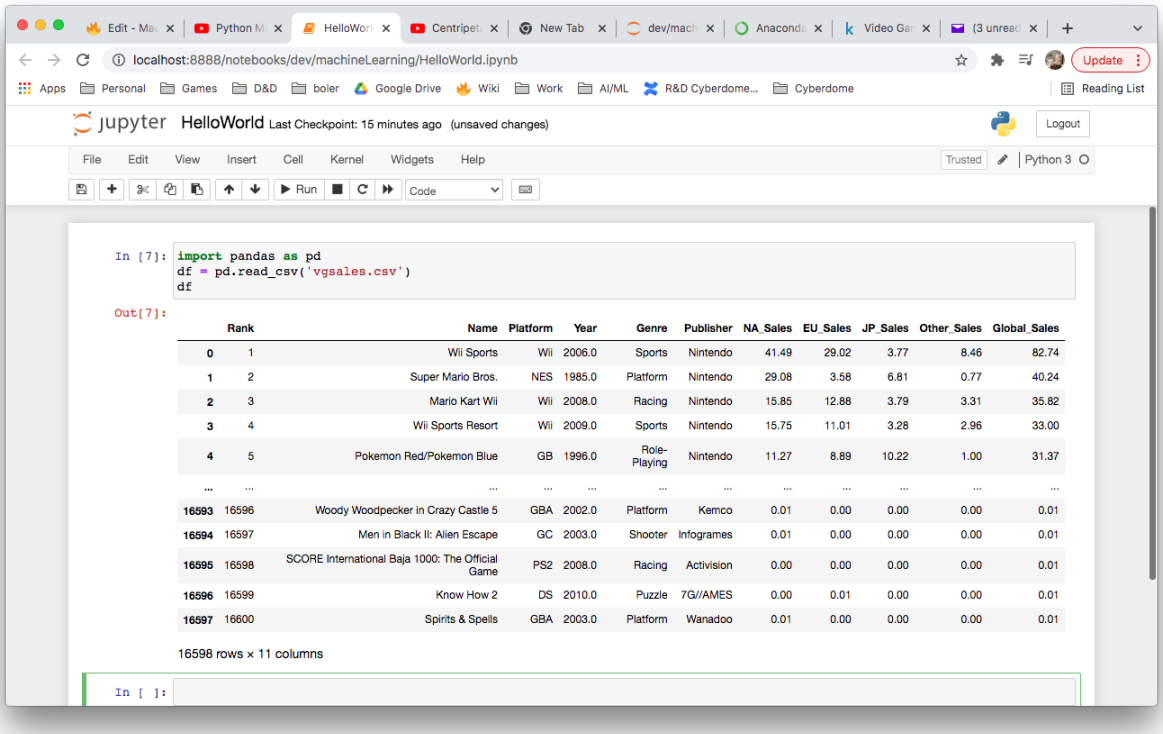
Import a Dataset

We can get some sample datasets from kaggle.com - https://www.kaggle.com/

From our Jupyter notebook, we are going to import a downloaded CSV.

```
import panda as pd
df = pd.read_csv('vgsales.csv')
df
```

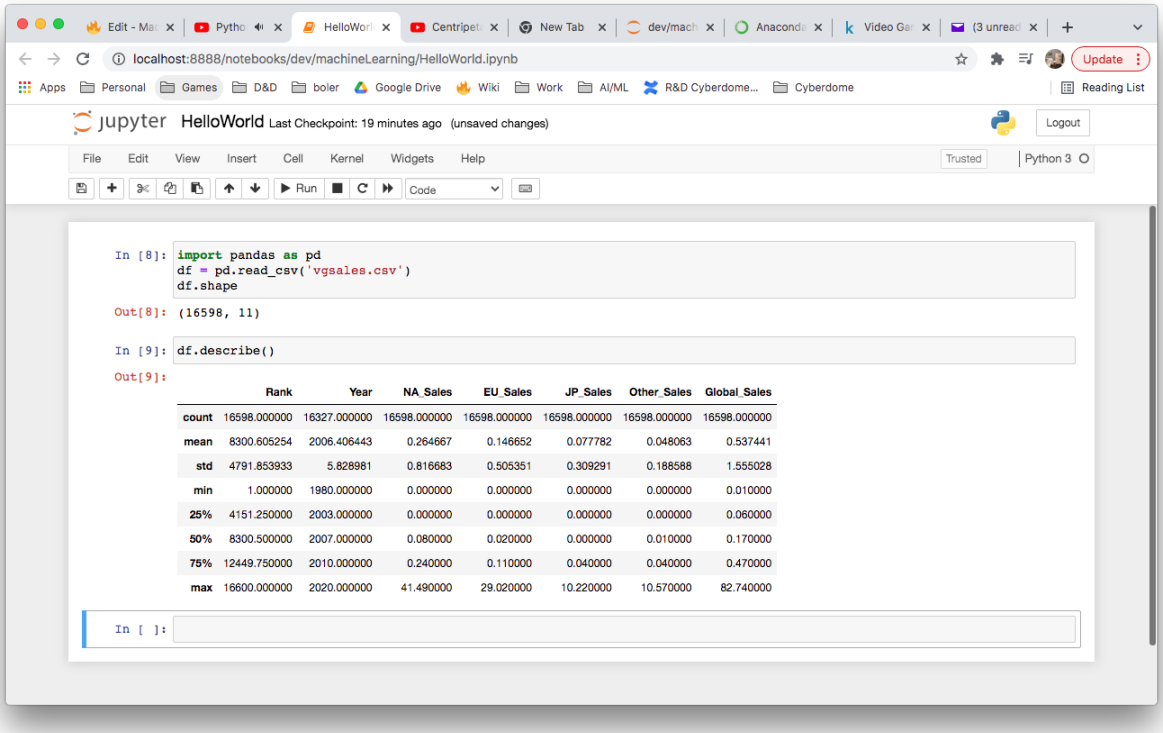
The pd.read function returns a **DataFrame** object



	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37
...	...	...	...	...	...	...	...	...	...	...	...
16593	16596	Woody Woodpecker in Crazy Castle 5	GBA	2002.0	Platform	Kemco	0.01	0.00	0.00	0.00	0.01
16594	16597	Men in Black II: Alien Escape	GC	2003.0	Shooter	Infogrames	0.01	0.00	0.00	0.00	0.01
16595	16598	SCORE International Baja 1000: The Official Game	PS2	2008.0	Racing	Activision	0.00	0.00	0.00	0.00	0.01
16596	16599	Know How 2	DS	2010.0	Puzzle	7G//AMES	0.00	0.01	0.00	0.00	0.01
16597	16600	Spirits & Spells	GBA	2003.0	Platform	Wanadoo	0.01	0.00	0.00	0.00	0.01

16598 rows × 11 columns

Dataframe Functions:



	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
count	16598.000000	16327.000000	16598.000000	16598.000000	16598.000000	16598.000000	16598.000000
mean	8300.605254	2006.406443	0.264667	0.146652	0.077782	0.048063	0.537441
std	4791.853933	5.828981	0.816683	0.505351	0.309291	0.188588	1.555028
min	1.000000	1980.000000	0.000000	0.000000	0.000000	0.000000	0.010000
25%	4151.250000	2003.000000	0.000000	0.000000	0.000000	0.000000	0.060000
50%	8300.500000	2007.000000	0.080000	0.020000	0.000000	0.010000	0.170000
75%	12449.750000	2010.000000	0.240000	0.110000	0.040000	0.040000	0.470000
max	16600.000000	2020.000000	41.490000	29.020000	10.220000	10.570000	82.740000

Interesting DataFrame functions:

Method	Description	Example
shape	returns dimensions of dataset	df.shape `(16598, 11)`
describe	returns useful statistics about our data	df.describe() (see above image)
values	returns your data	

Jupyter Shortcuts

Shortcut	Mode	Key	Description
Add Cell Above	Command	a	
Add Cell Below	Command	b	
Delete Current Cell	Command	dd	
Run current Cell and Stay in Cell	Command/Edit	<CTRL><ENTER>	Run Commands in cell without adding a cell below.
Autocompletion	Edit	<TAB>	Get methods for object
Method Documentation	Edit	<SHIFT> <TAB>	Get information on method
Make Comment	Edit	<CMD> /	Comment/UnComment

Real Example

Import the data

```
import pandas as pd
df = pd.read_csv('music.csv')
df
```

Spit the Data

Create input and output data sets. X = input, y = output.

Since we want to predict the type of music based on age and sex, we create our input data as X and our output as y.

```
import pandas as pd
df = pd.read_csv('music.csv')
X = df.drop(columns="genre")
y = df["genre"]
y
```

Train and Do a Prediction

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier

df = pd.read_csv('music.csv')
X = df.drop(columns="genre")
y = df["genre"]

model = DecisionTreeClassifier()

# train model
model.fit(X,y)

# predict
# 21 year old male and 22 year old female
predictions = model.predict([[21,1],[22,0]])
predictions
```

In the above example, we used 100% of the data for training and 0 for testing our model.

Testing our Model

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

df = pd.read_csv('music.csv')
X = df.drop(columns="genre")
y = df["genre"]

#split our data into train and test DataFrames (20% for testing)
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2)

model = DecisionTreeClassifier()

# train model
model.fit(X_train,y_train)

# run predict using test data
predictions = model.predict(X_test)
score = accuracy_score(y_test, predictions)
score
```

```
In [25]: import pandas as pd
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score

         df = pd.read_csv('music.csv')
         X = df.drop(columns="genre")
         y = df["genre"]

         #split our data into train and test DataFrames (20% for testing)
         X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2)

         model = DecisionTreeClassifier()

         # train model
         model.fit(X_train,y_train)

         # run predict using test data
         predictions = model.predict(X_test)
         score = accuracy_score(y_test, predictions)
         score

Out[25]: 0.75
```

Model Persistence

Saving a Trained Model

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import joblib

df = pd.read_csv('music.csv')
X = df.drop(columns="genre")
y = df["genre"]

#split our data into train and test DataFrames (20% for testing)
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2)

model = DecisionTreeClassifier()

# train model
model.fit(X_train,y_train)

# run predict using test data
# predictions = model.predict(X_test)
# score = accuracy_score(y_test, predictions)

#save our model
joblib.dump(model,"music-recomender.joblib")
```

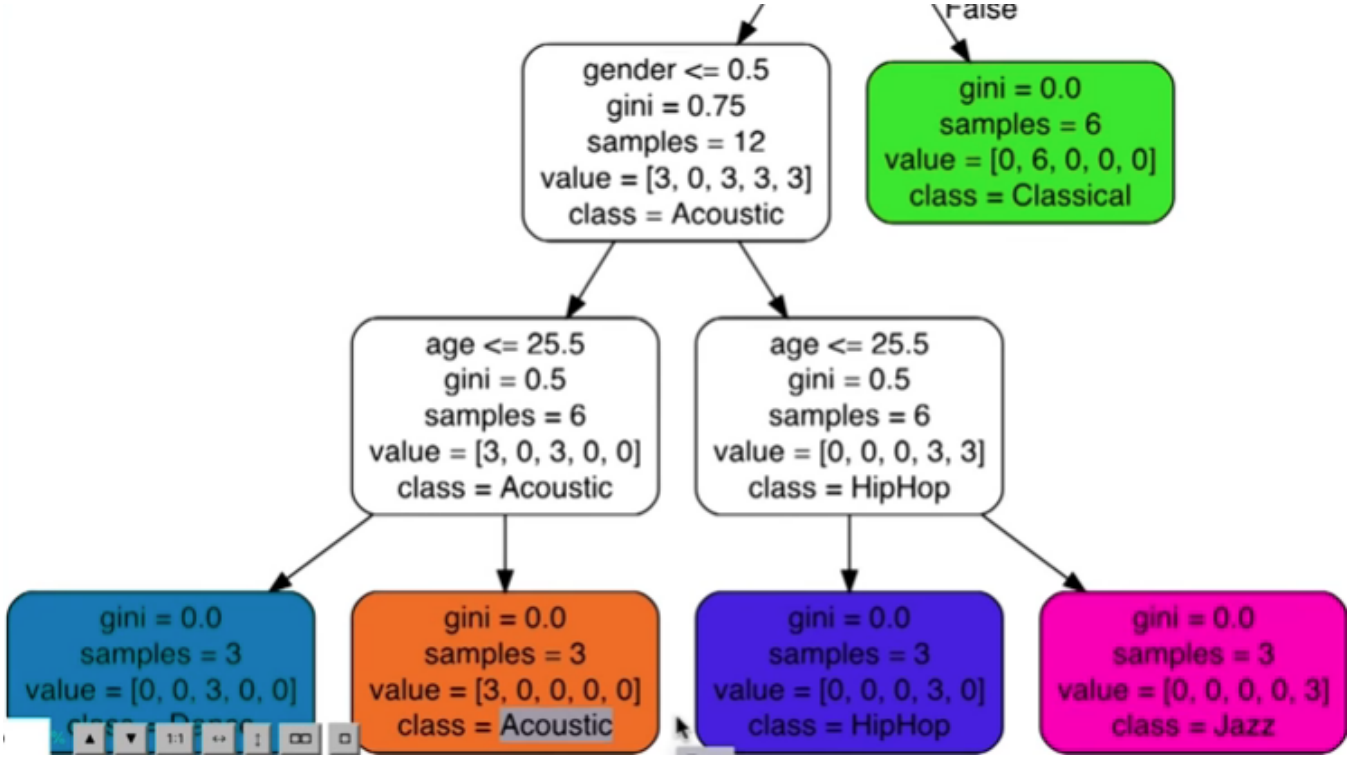
Predictions from a Saved Model

```
import joblib

#load our model
model = joblib.load("music-recomender.joblib")

# run predict using test data
predictions = model.predict([[20,1],[21,0]])

predictions
```

Visualizing Decision Trees

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree

df = pd.read_csv('music.csv')
X = df.drop(columns="genre")
y = df["genre"]

model = DecisionTreeClassifier()

# train model
model.fit(X,y)

# export graph of data in dot format
tree.export_graphviz(model,out_file='music_recomender.dot',
                     feature_names=['age','gender'],
                     class_names=sorted(y.unique()),
                     label='all',
                     rounded=True,
                     filled=True)
```

This will output our .dot file. We just need to pull it into VSCode with dot plugin to visualize it.



References

Reference	URL
Python Machine Learning Tutorial (Data Science)	https://www.youtube.com/watch?v=7eh4d6sabA0