



Machine Learning

www.hbpatel.in



Applications of Machine Learning

www.hbpatel.in

Applications of Machine Learning

- Self-driving vehicles
- Robotics
- Language Processing
- Vision Processing
- Stock Market Prediction



Steps involved in Machine Learning

www.hbpatel.in

Steps of Machine Learning Project

- Import Data
- Clean Data
- Split Data (Training & Testing sets)
- Create a Model
- Train the Model
- Make Predictions
- Evaluate and Improve



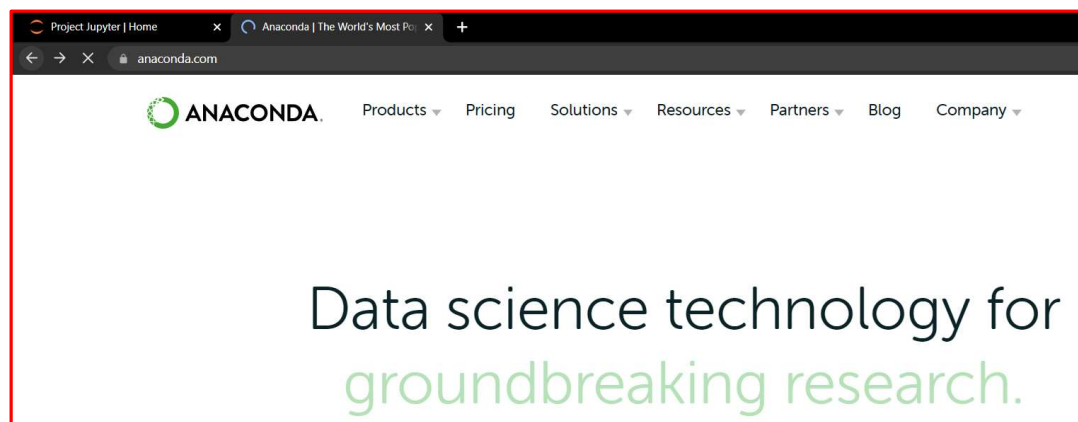
Machine Learning (Libraries and Tools)

www.hbpatel.in

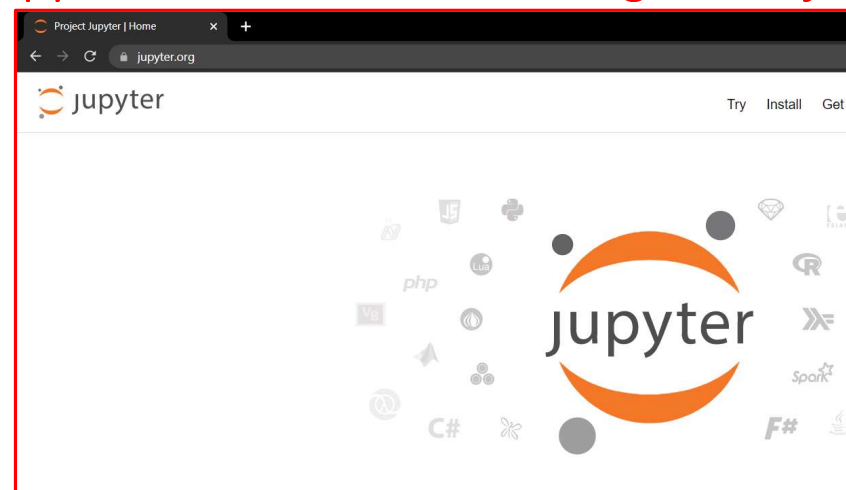
Libraries

- numpy
- Pandas
- Matplotlib
- Scikit-learn

To install Jupyter, we use the platform of Anaconda



Jupyter Environment for Writing ML Project





Step 1: Import Data

www.hbpatel.in

AutoSave ☐ Off music

File Home Insert Page Layout

Undo Paste Cut Copy Format Painter

Clipboard

A1 age

	A	B	C	D
1	age	gender	genre	
2	20	1	HipHop	
3	23	1	HipHop	
4	25	1	HipHop	
5	26	1	Jazz	
6	29	1	Jazz	
7	30	1	Jazz	
8	31	1	Classical	
9	33	1	Classical	
10	37	1	Classical	
11	20	0	Dance	
12	21	0	Dance	
13	25	0	Dance	
14	26	0	Acoustic	
15	27	0	Acoustic	
16	30	0	Acoustic	
17	31	0	Classical	
18	34	0	Classical	
19	35	0	Classical	
20				

HelloWorld - Jupyter Notebook | Video Game Sales | Kaggle | Dropbox - 404

localhost:8888/tree/Desktop

jupyter

Files Running Clusters

Select items to perform actions on them.

0 / Desktop

- HelloWorld.ipynb
- music.csv
- vgsales.csv

```
In [11]: import pandas as pd
music_data = pd.read_csv('music.csv')
music_data
```

```
Out[11]:
```

	age	gender	genre
0	20	1	HipHop
1	23	1	HipHop
2	25	1	HipHop
3	26	1	Jazz
4	29	1	Jazz
5	30	1	Jazz
6	31	1	Classical
7	33	1	Classical
8	37	1	Classical
9	20	0	Dance
10	21	0	Dance
11	25	0	Dance
12	26	0	Acoustic
13	27	0	Acoustic
14	30	0	Acoustic
15	31	0	Classical
16	34	0	Classical

Upload New

Name	Last Modified	File size
	seconds ago	
Running	5 hours ago	589 B
	a minute ago	270 B
	5 hours ago	1.36 MB



Step 2: Clean Data

www.hbpatel.in

Remove duplicates and clean NULL values but we don't have such thing in our current music.csv file. However, we may split the database into two table, first containing the input (first two columns) and second containing the output (last column).

```
In [12]: import pandas as pd
music_data = pd.read_csv('music.csv')
x = music_data.drop(columns=['genre'])
x
```

```
Out[12]:
```

	age	gender
0	20	1
1	23	1
2	25	1
3	26	1
4	29	1
5	30	1
6	31	1
7	33	1
8	37	1
9	20	0
10	21	0
11	25	0
12	26	0
13	27	0
14	30	0
15	31	0

```
In [19]: import pandas as pd
music_data = pd.read_csv('music.csv')
x = music_data.drop(columns=['genre'])
y = music_data['genre']
y
```

```
Out[19]:
```

0	HipHop
1	HipHop
2	HipHop
3	Jazz
4	Jazz
5	Jazz
6	Classical
7	Classical
8	Classical
9	Dance
10	Dance
11	Dance
12	Acoustic
13	Acoustic
14	Acoustic
15	Classical
16	Classical
17	Classical

Name: genre, dtype: object



Step 3: Learning and Predicting Data

www.hbpatel.in

```
In [43]: import pandas as pd
from sklearn.tree import DecisionTreeClassifier

#import dataset
music_data = pd.read_csv('music.csv')
#create input set
X = music_data.drop(columns=['genre'])
#create output set
y = music_data['genre']

#create a model
model = DecisionTreeClassifier()
#train the model
model.fit(X, y)

music_data
```

Out[43]:

	age	gender	genre
0	20	1	HipHop
1	23	1	HipHop
2	25	1	HipHop
3	26	1	Jazz
4	29	1	Jazz
5	30	1	Jazz
6	31	1	Classical
7	33	1	Classical

```
In [25]: import pandas as pd
from sklearn.tree import DecisionTreeClassifier

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

model = DecisionTreeClassifier()
model.fit(col1, col2)

predictions = model.predict([ [21, 1], [22, 0] ])
predictions
```

Out[25]: array(['HipHop', 'Dance'], dtype=object)

In []:

Q: What does a male (1) of age (21) like? [21, 1]
Prediction: HipHop

Q: What does a female (0) of age (22) like? [22, 0]
Prediction: Dance



Step 4: Calculating Accuracy

www.hbpatel.in

Divide the dataset into two sets viz. (A) training dataset (70-80%), (B) testing dataset (20-30%)

Instead of passing one or two samples, we'll pass testing dataset and compare the resultant predictions with the actual values. Based on that, we can calculate the accuracy.

`accuracy_score(y_test, predictions)`: `y_test`: Expected values, `predictions`: Actual values.

```
In [26]: import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

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

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

model = DecisionTreeClassifier()
model.fit(X_train, y_train)
|
predictions = model.predict(X_test)

score = accuracy_score(y_test, predictions)
score
```

Out[26]: 1.0

You may run this again and again with different size of training/testing data (E.g. `test_size=0.5`)

Out[29]: 0.75



Step 5: Persisting Models

www.hbpatel.in

```
In [39]: import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.externals import joblib

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

model = DecisionTreeClassifier()
model.fit(X, y)

joblib.dump(model, 'music-recommender.joblib')
```

```
Out[39]: ['music-recommender.joblib']
```

Quit Logout

Select items to perform actions on them.

Upload New ↺

<input type="checkbox"/> 0	Name ↓	Last Modified	File size
	..	seconds ago	
<input type="checkbox"/>	HelloWorld.ipynb	Running 2 minutes ago	1.12 kB
<input type="checkbox"/>	music-recommender.joblib	a minute ago	2.54 kB
<input type="checkbox"/>	music.csv	2 hours ago	270 B
<input type="checkbox"/>	vgsales.csv	7 hours ago	1.36 MB



Step 5: Persisting Models

www.hbpatel.in

```
In [41]: import pandas as pd
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.externals import joblib

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

         # model = DecisionTreeClassifier()
         # model.fit(X, y)

         model = joblib.load('music-recommender.joblib')
         predictions = model.predict([[21, 1]])
         predictions
```

```
Out[41]: array(['HipHop'], dtype=object)
```



Step 6: Visualizing a Decision Tree

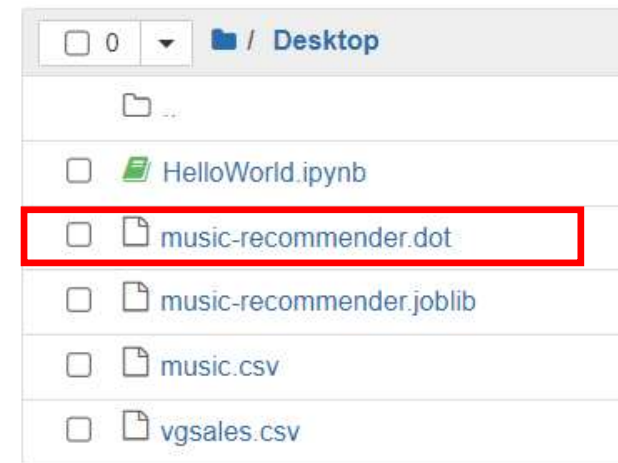
www.hbpatel.in

```
In [45]: import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree

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

model = DecisionTreeClassifier()
model.fit(X, y)

tree.export_graphviz(model, out_file='music-recommender.dot',
                      feature_names=['age', 'gender'],
                      class_names=sorted(y.unique()),
                      label='all',
                      rounded=True,
                      filled=True)
```





Step 6: Visualizing a Decision Tree

www.hbpatel.in

Open the music-recommender.dot [from C:\Users\Hiren Patel\Desktop] file in Visual Studio Code.

```
1 digraph Tree {
2   node [shape=box, style="filled, rounded", color="black", fontname="helvetica"] ;
3   edge [fontname="helvetica"] ;
4   0 [label="age <= 30.5\ngini = 0.778\nsamples = 18\nvalue = [3, 6, 3, 3, 3]\nclass = Classical", fillcolor="#e5fad7"] ;
5   1 [label="age <= 25.5\ngini = 0.75\nsamples = 12\nvalue = [3, 0, 3, 3, 3]\nclass = Acoustic", fillcolor="ffffff"] ;
6   0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
7   2 [label="gender <= 0.5\ngini = 0.5\nsamples = 6\nvalue = [0, 0, 3, 3, 0]\nclass = Dance", fillcolor="ffffff"] ;
8   1 -> 2 ;
9   3 [label="gini = 0.0\nsamples = 3\nvalue = [0, 0, 3, 0, 0]\nclass = Dance", fillcolor="#39e5c5"] ;
10  2 -> 3 ;
11  4 [label="gini = 0.0\nsamples = 3\nvalue = [0, 0, 0, 3, 0]\nclass = HipHop", fillcolor="#3c39e5"] ;
12  2 -> 4 ;
13  5 [label="gender <= 0.5\ngini = 0.5\nsamples = 6\nvalue = [3, 0, 0, 0, 3]\nclass = Acoustic", fillcolor="ffffff"] ;
14  1 -> 5 ;
15  6 [label="gini = 0.0\nsamples = 3\nvalue = [3, 0, 0, 0, 0]\nclass = Acoustic", fillcolor="#e58139"] ;
16  5 -> 6 ;
17  7 [label="gini = 0.0\nsamples = 3\nvalue = [0, 0, 0, 0, 3]\nclass = Jazz", fillcolor="#e539c0"] ;
18  5 -> 7 ;
19  8 [label="gini = 0.0\nsamples = 6\nvalue = [0, 6, 0, 0, 0]\nclass = Classical", fillcolor="#7be539"] ;
20  0 -> 8 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
21 }
```



Step 6: Visualizing a Decision Tree

www.hbpatel.in

The screenshot shows the Visual Studio Code interface with the Extensions Marketplace open. The search bar contains the text "dot". The extension "Graphviz (dot) language support for Visual Studio Code" by João Pinto is highlighted in the list on the left. The main panel displays the details for this extension, including its version (v0.0.6), a rating of 5 stars, and an "Install" button. The extension's description states: "This extension provides GraphViz (dot) language support for Visual Studio Code". The "Details" tab is selected, showing a "Graphviz Support" section with a description: "A vscode extension that provides language support and live preview for the Graphviz format." It also mentions that the preview uses the Viz.js library and that the extension can be activated in two ways. The "Features" section lists snippets for creating graphs, variables, properties, paths, and ranks. The "Graph preview" section includes instructions on how to toggle the preview and open it to the side. The right sidebar shows categories (Programming Languages, Snippets), extension resources (Marketplace, Repository, License, João Pinto), and more info (Released on, Last updated, Identifier).

Visual Studio Code interface showing the installation of the **Graphviz (dot) language support for Visual Studio Code** extension.

The extension is developed by **João Pinto** and has a rating of 5 stars (12 reviews).

The extension provides **GraphViz (dot) language support for Visual Studio Code**.

Graphviz Support

A vscode extension that provides language support and live preview for the Graphviz format.

The preview uses the [Viz.js](#) library.

The extension can be activated in two ways

Features

Snippets

Try typing one of the following prefixes to see available snippets: graph, >, var, dir, prop, path or rank and efficiently create graphs, variables, properties, paths or ranks.

Graph preview

- Toggle Preview - `ctrl+shift+v` (Mac: `cmd+shift+v`)
- Open Preview to the Side - `ctrl+k v` (Mac: `cmd+k shift+v`)

Categories

- Programming Languages
- Snippets

Extension Resources

- Marketplace
- Repository
- License
- João Pinto

More Info

- Released on: 3/26/2018, 02:22:37
- Last updated: 2/26/2020, 15:46:02
- Identifier: joaompinto.vscode-graphviz



Step 6: Visualizing a Decision Tree

www.hbpatel.in

Reload Visual Studio Code.

```
1 digraph Tree {
2 node [shape=box, style="filled, rounded", color="black", fontname="helvetica"] ;
3 edge [fontname="helvetica"] ;
4 0 [label="age <= 30.5\ngini = 0.778\nsamples = 18\nvalue = [3, 6, 3, 3, 3]\nclass = Classical", fillcolor="#e5f5e0"]
5 1 [label="age <= 25.5\ngini = 0.75\nsamples = 12\nvalue = [3, 0, 3, 3, 3]\nclass = Acoustic", fillcolor="#ffffcc"]
6 0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
7 2 [label="gender <= 0.5\ngini = 0.5\nsamples = 6\nvalue = [0, 0, 3, 3, 0]\nclass = Dance", fillcolor="white"]
8 1 -> 2 ;
9 3 [label="gini = 0.0\nsamples = 3\nvalue = [0, 0, 3, 0, 0]\nclass = Dance", fillcolor="#39e5c5"] ;
10 2 -> 3 ;
11 4 [label="gini = 0.0\nsamples = 3\nvalue = [0, 0, 0, 3, 0]\nclass = HipHop", fillcolor="#3c39e5"] ;
12 2 -> 4 ;
13 5 [label="gender <= 0.5\ngini = 0.5\nsamples = 6\nvalue = [3, 0, 0, 0, 3]\nclass = Acoustic", fillcolor="white"] ;
14 1 -> 5 ;
15 6 [label="gini = 0.0\nsamples = 3\nvalue = [3, 0, 0, 0, 0]\nclass = Acoustic", fillcolor="#e58139"] ;
16 5 -> 6 ;
17 7 [label="gini = 0.0\nsamples = 3\nvalue = [0, 0, 0, 0, 3]\nclass = Jazz", fillcolor="#e539c0"] ;
18 5 -> 7 ;
19 8 [label="gini = 0.0\nsamples = 6\nvalue = [0, 6, 0, 0, 0]\nclass = Classical", fillcolor="#7be539"] ;
20 0 -> 8 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
21 }
```



Step 6: Visualizing a Decision Tree

www.hbpatel.in

