DS-210: Programming for Data Science

Lecture 5:

- Follow-up on Pandas dtypes
- Classification vs. regression
- Ethics of data processing

Announcements

Midterm: 02/28, 12:20–1:10pm, MCS B37
Final: 05/13, 12–2pm, MCS B37

- Both open book
- Midterm: focus on Python + machine learning
- Final: focus on Rust + basic algorithms and data structures

Announcements

Midterm: 02/28, 12:20–1:10pm, MCS B37
Final: 05/13, 12–2pm, MCS B37

- Both open book
- Midterm: focus on Python + machine learning
- Final: focus on Rust + basic algorithms and data structures

Gradescope

- Entry code: 3Y85PZ
- Can you submit an IPython notebook? Or a `.zip` file?

Setting up our environment for this lecture

In [1]:
```
import numpy as np
import pandas as pd
import random
import math
from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor
from sklearn import tree
import matplotlib.pyplot as plt

# print a text file
def print_file(filename):
    with open(filename) as f:
        print(f.read(),end='')
    print("================")

# display data we have read
def show_data(data):
    print(data.dtypes,data,sep='\n---\n')
```

Follow-up: string dtypes

In [2]:
```
print_file('data.csv')
data = pd.read_csv("data.csv")
show_data(data)
```

```
Name,LikesIceCream,Age
Alice,True,54
Bob,False,52
Carol,False,44
Eugene,True,61
================
Name             object
LikesIceCream      bool
Age               int64
dtype: object
---
     Name  LikesIceCream  Age
0   Alice           True   54
1     Bob          False   52
2   Carol          False   44
3  Eugene           True   61
```

Follow-up: string dtypes

In [2]:
```
print_file('data.csv')
data = pd.read_csv("data.csv")
show_data(data)
```

```
Name,LikesIceCream,Age
Alice,True,54
Bob,False,52
Carol,False,44
Eugene,True,61
================
Name             object
LikesIceCream      bool
Age               int64
dtype: object
---
     Name  LikesIceCream  Age
0   Alice           True   54
1     Bob          False   52
2   Carol          False   44
3  Eugene           True   61
```

In [3]:
```
types = {'Name':'string','LikesIceCream':'string'}
data = pd.read_csv("data.csv",dtype=types)
show_data(data)
```

```
Name             string
LikesIceCream    string
Age               int64
dtype: object
---
     Name LikesIceCream  Age
0   Alice           True   54
1     Bob          False   52
2   Carol          False   44
3  Eugene           True   61
```

Follow-up: string dtypes

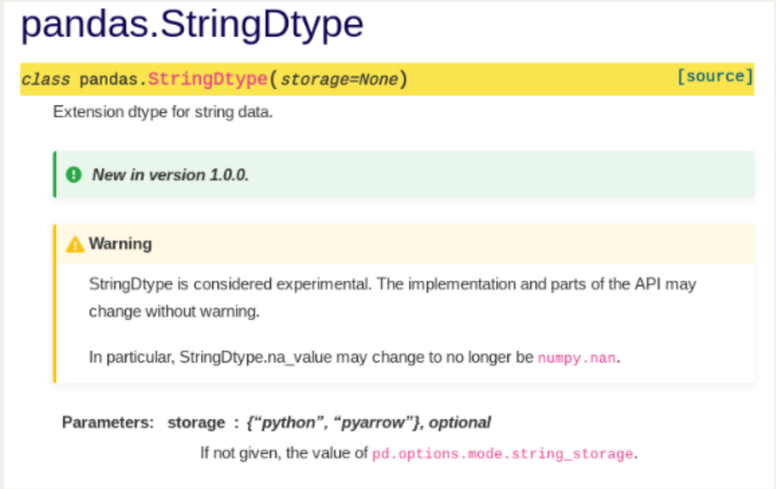
In [2]:
```
print_file('data.csv')
data = pd.read_csv("data.csv")
show_data(data)
```

```
Name,LikesIceCream,Age
Alice,True,54
Bob,False,52
Carol,False,44
Eugene,True,61
================
Name             object
LikesIceCream      bool
Age               int64
dtype: object
---
     Name  LikesIceCream  Age
0   Alice           True   54
1     Bob          False   52
2   Carol          False   44
3  Eugene           True   61
```

In [3]:
```
types = {'Name':'string','LikesIceCream':'string'}
data = pd.read_csv("data.csv",dtype=types)
show_data(data)
```

```
Name             string
LikesIceCream    string
Age               int64
dtype: object
---
     Name LikesIceCream  Age
0   Alice           True   54
1     Bob          False   52
2   Carol          False   44
3  Eugene           True   61
```

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.StringDtype.html



pandas.StringDtype

class pandas.StringDtype(*storage=None*) [source]

Extension dtype for string data.

New in version 1.0.0.

Warning

StringDtype is considered experimental. The implementation and parts of the API may change without warning.

In particular, StringDtype.na_value may change to no longer be numpy.nan.

Parameters: **storage** : *{"python", "pyarrow"}, optional*

If not given, the value of pd.options.mode.string_storage.

Follow-up: other dtypes

In [4]:
```
print_file('data2.csv')
data = pd.read_csv("data2.csv")
show_data(data)
```

```
Name,LikesIceCream,Age
Alice,True,54
Bob,False,52
Carol,False,44
Eugene,True,130
================
Name             object
LikesIceCream      bool
Age               int64
dtype: object
---
     Name  LikesIceCream  Age
0   Alice           True   54
1     Bob          False   52
2   Carol          False   44
3  Eugene           True  130
```

In [5]:
```
types = {'Age':'int8'}
data = pd.read_csv("data2.csv",dtype=types)
show_data(data)
```

```
Name             object
LikesIceCream      bool
Age                int8
dtype: object
---
     Name  LikesIceCream  Age
0   Alice           True   54
1     Bob          False   52
2   Carol          False   44
3  Eugene           True -126
```

Warning: no error when an input integer overflows a small–size integer variable!

Follow-up: other dtypes

https://pandas.pydata.org/pandas-docs/stable/reference/arrays.html

Kind of Data	pandas Data Type	Scalar	Array
TZ-aware datetime	`DatetimeTZDtype`	`Timestamp`	Datetime data
Timedeltas	(none)	`Timedelta`	Timedelta data
Period (time spans)	`PeriodDtype`	`Period`	Timespan data
Intervals	`IntervalDtype`	`Interval`	Interval data
Nullable Integer	`Int64Dtype`, ...	(none)	Nullable integer
Categorical	`CategoricalDtype`	(none)	Categorical data
Sparse	`SparseDtype`	(none)	Sparse data
Strings	`StringDtype`	`str`	Text data
Boolean (with NA)	`BooleanDtype`	`bool`	Boolean data with missing values

Terminology: Classification vs. regression

Terminology: Classification vs. regression

Classification:

- **Possible answers:** (small) finite set of options
- **Example 1:** `cat`, `dog`, `cow`
- **Example 2:** `True` or `False`
- **Typical success evaluation:** fraction of samples with correct prediction

Terminology: Classification vs. regression

Classification:

- **Possible answers:** (small) finite set of options
- **Example 1:** `cat`, `dog`, `cow`
- **Example 2:** `True` or `False`
- **Typical success evaluation:** fraction of samples with correct prediction

Regression:

- **Possible answers:** real numbers
- **Example:** approximate the likelihood of cancer
- **Popular success evaluation:**

$$\frac{1}{|S|} \sum_{(x,y) \in S} (f(x) - y)^2$$

where

  - f is a regressor
  - S is a test set

Terminology: Classification vs. regression

Classification:

- **Possible answers:** (small) finite set of options
- **Example 1:** `cat`, `dog`, `cow`
- **Example 2:** `True` or `False`
- **Typical success evaluation:** fraction of samples with correct prediction

Regression:

- **Possible answers:** real numbers
- **Example:** approximate the likelihood of cancer
- **Popular success evaluation:**

$$\frac{1}{|S|} \sum_{(x,y) \in S} (f(x) - y)^2$$

where

  - f is a regressor
  - S is a test set

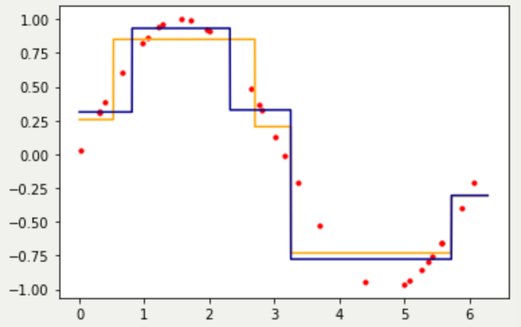
Which is the right choice for Homework 1?

sklearn.tree.DecisionTree{Classifier,Regressor}

- Use `DecisionTreeClassifier` for classification
- Use `DecisionTreeRegressor` for regression

In [6]:
```
# Data preparation
samples = 30
leafs = 5
X = [[random.uniform(0,2*math.pi)]\
     for i in range(samples)]
y = [math.sin(x[0]) for x in X]

# Training a decision tree regressor
reg = DecisionTreeRegressor(max_leaf_nodes=leafs)
reg = reg.fit(X,y)

# Same thing with a different target error
reg_2 = DecisionTreeRegressor(max_leaf_nodes=leafs,\
        criterion='absolute_error')
reg_2 = reg_2.fit(X,y)

# prepare data for plotting
plot_x = np.arange(0,2*math.pi,0.001)
plot_y = reg.predict([[x] for x in plot_x])
plot_y_2 = reg_2.predict([[x] for x in plot_x])
```

In [7]:
```
plt.scatter(X,y,s=10,c='red')
plt.plot(plot_x,plot_y,color='orange');
plt.plot(plot_x,plot_y_2,color='darkblue');
```



Decision area visualization (simple tool)

- See the attached `decision_area.py`
- Very basic, feel free to modify it

```
In [8]: data = pd.read_csv("pizza.csv"); data
```

Out[8]:

	Name	Number	PPG	YearBorn	TotalPoints	LikesPizza
0	Kareem	33	24.6	1947	38387	1
1	Karl	32	25.0	1963	36928	0
2	LeBron	23	27.0	1984	36381	0
3	Kobe	24	25.0	1978	33643	1
4	Michael	23	30.1	1963	32292	0

Decision area visualization (simple tool)

- See the attached `decision_area.py`
- Very basic, feel free to modify it

In [8]:
```
data = pd.read_csv("pizza.csv"); data
```

Out[8]:

	Name	Number	PPG	YearBorn	TotalPoints	LikesPizza
0	Kareem	33	24.6	1947	38387	1
1	Karl	32	25.0	1963	36928	0
2	LeBron	23	27.0	1984	36381	0
3	Kobe	24	25.0	1978	33643	1
4	Michael	23	30.1	1963	32292	0

In [9]:
```
# select features
features = ['PPG','YearBorn','TotalPoints']
X = data[features]
y = data['LikesPizza']

# train classifier
clf = DecisionTreeClassifier(max_leaf_nodes = 4,\
                             random_state=0)
clf = clf.fit(X,y)

# export to text form
txt = tree.export_text(clf,feature_names = features)
```

Decision area visualization (simple tool)

- See the attached `decision_area.py`
- Very basic, feel free to modify it

In [8]:
```
data = pd.read_csv("pizza.csv"); data
```

Out[8]:

	Name	Number	PPG	YearBorn	TotalPoints	LikesPizza
0	Kareem	33	24.6	1947	38387	1
1	Karl	32	25.0	1963	36928	0
2	LeBron	23	27.0	1984	36381	0
3	Kobe	24	25.0	1978	33643	1
4	Michael	23	30.1	1963	32292	0

In [9]:
```
# select features
features = ['PPG','YearBorn','TotalPoints']
X = data[features]
y = data['LikesPizza']

# train classifier
clf = DecisionTreeClassifier(max_leaf_nodes = 4,\
                             random_state=0)
clf = clf.fit(X,y)

# export to text form
txt = tree.export_text(clf,feature_names = features)
```

In [10]:
```
print(txt)
```

```
|--- PPG <= 26.00
|   |--- TotalPoints <= 35285.50
|   |   |--- class: 1
|   |--- TotalPoints >  35285.50
|   |   |--- PPG <= 24.80
|   |   |   |--- class: 1
|   |   |--- PPG >  24.80
|   |   |   |--- class: 0
|--- PPG >  26.00
|   |--- class: 0
```



Decision area visualization (simple tool)

- See the attached `decision_area.py`
- Very basic, feel free to modify it

In [8]:
```
data = pd.read_csv("pizza.csv"); data
```

Out[8]:

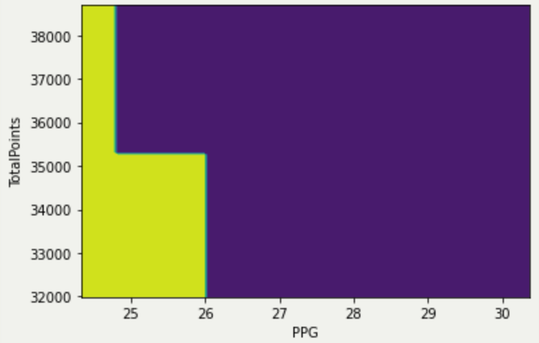
	Name	Number	PPG	YearBorn	TotalPoints	LikesPizza
0	Kareem	33	24.6	1947	38387	1
1	Karl	32	25.0	1963	36928	0
2	LeBron	23	27.0	1984	36381	0
3	Kobe	24	25.0	1978	33643	1
4	Michael	23	30.1	1963	32292	0

In [9]:
```
# select features
features = ['PPG','YearBorn','TotalPoints']
X = data[features]
y = data['LikesPizza']

# train classifier
clf = DecisionTreeClassifier(max_leaf_nodes = 4,\
                             random_state=0)
clf = clf.fit(X,y)

# export to text form
txt = tree.export_text(clf,feature_names = features)
```

In [10]:
```
print(txt)
```

```
|--- PPG <= 26.00
|   |--- TotalPoints <= 35285.50
|   |   |--- class: 1
|   |--- TotalPoints >  35285.50
|   |   |--- PPG <= 24.80
|   |   |   |--- class: 1
|   |   |--- PPG >  24.80
|   |   |   |--- class: 0
|--- PPG >  26.00
|   |--- class: 0
```

In [11]:
```
from decision_area import draw_decision_area
draw_decision_area(clf,X,'PPG','TotalPoints')
```



Ethics of data processing

- **Lots of data sets have private information**
- **Most infamous examples:**
    - Enron emails
    - AOL search
    - Netflix data set
- **Bottom line:**
    - Be careful publishing any data
    - Respect privacy of subjects

Ethics of data processing

- **Lots of data sets have private information**
- **Most infamous examples:**
    - Enron emails
    - AOL search
    - Netflix data set
- **Bottom line:**
    - Be careful publishing any data
    - Respect privacy of subjects

Next time

- Will discuss the expectations for the final project