🐣 Machine Learing - Sklearn

basic ML workflow

1. split data
2. train model
3. score
4. evaluate

```
## sklearn basics
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
import pandas as pd
```

```
## read data
mtcars = pd.read_csv("https://gist.githubusercontent.com/seankross/a412dfbd88b3db70b74b/raw/5f23f993cd87c283ce766e7
mtcars.head()
```

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

Next steps: Generate code with mtcars | View recommended plots | New interactive sheet

```
## prepare data
X = mtcars.drop(["model", "mpg"], axis = 1) ## axis = 1 (column)
y = mtcars["mpg"]
```

```
## split data
## random_state = 42 (set.seed(42))
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)
```

```
## train model
model = LinearRegression()
model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
## prediction (score)
pred = model.predict(X_test)
```

```
## R squared (evaluate)
model.score(X_test, y_test)
```

```
0.7856209608689562
```

```
model.score(X_train, y_train)
```

```
0.8667068951242609
```

🎯 RandomForestRegressor

```
from sklearn.ensemble import RandomForestRegressor
```

```
## train model
model = RandomForestRegressor()
model.fit(X_train, y_train)
```

```
▾ RandomForestRegressor
RandomForestRegressor()
```

```python
## prediction (score)
pred = model.predict(X_test)
```

```python
## R squared (evaluate)
model.score(X_test, y_test)
```

```
0.8862972637723459
```

```python
model.score(X_train, y_train)
```

```
0.9699666651769879
```

🎯 DecisionTreeRegressor

```python
from sklearn.tree import DecisionTreeRegressor
```

```python
## train model
model = DecisionTreeRegressor()
model.fit(X_train, y_train)
```

```
▾ DecisionTreeRegressor
DecisionTreeRegressor()
```

```python
## prediction (score)
pred = model.predict(X_test)
```

```python
## R squared (evaluate)
model.score(X_test, y_test)
```

```
0.8550103368600268
```

```python
model.score(X_train, y_train)
```

```
1.0
```