

Good features beat algorithms

Pycon X - Firenze 

3 May 2019

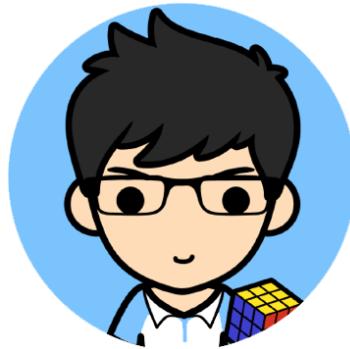
Pietro Mascolo
Director of Machine Learning - Flow Commerce

Slides and code: bit.ly/2MPPg5j (<https://bit.ly/2MPPg5j>)

I'm going to:

- introduce myself;
- tell a story;
- show you some code;
- show you some numbers.

Me



- (*) + + ;
- Physicist → Data Scientist → ML/AI;
- **Python / Go / Kotlin / Scala / ... ;**
- + +
- ...

Who we are



- Funded in 2015;
- Based in Hoboken 🇺🇸 and Dublin 🇮🇪;
- ~ 50 people;

Flow provides a turn-key platform for e-commerce companies to go global.

Global e-commerce now has Flow!

www.flow.io

4

What we do...

Global e-commerce is **HUGE**:

23% YoY growth, 29% of total B2C commerce

Brands optimize global experience with Flow's platform

The image displays a screenshot of the Flow platform's user interface, which is designed to help brands manage their global e-commerce operations. The interface is organized into five main sections, each represented by a card:

- Product Localization:** Shows a mobile device screen displaying a product page for a "FOXY SHERPA JACKET" from "CARBON". The page includes a large image of a model wearing the jacket, a color selection dropdown (Black), a "SELECT SIZE" dropdown, an "ADD TO BAG" button, and a note about VAT and Duty.
- Landed Cost:** Shows a screen for selecting shipping methods. It lists two options under "DDP" (Pay Tax / Duty now):
 - DHL Express (1-3 Business Days) - Estimated delivery date: 18/06/11 - 18/06/13, CN¥15.00
 - SF Express (4-6 Business Days) - Estimated delivery date: 18/06/14 - 18/06/16, CN¥9.00
- Carrier Management:** Shows a "SHIPPING METHOD" section with two options under "DDU" (Pay Tax / Duty upon delivery):
 - DHL Express (1-3 Business Days) - Estimated delivery date: 18/06/11 - 18/06/13, CN¥20.00
 - SF Express (4-6 Business Days) - Estimated delivery date: 18/06/14 - 18/06/16, CN¥9.00
- Localized Checkout:** Shows a "PAYMENT METHOD" section with four options:
 - Credit Card
 - 中国银行 (Bank of China)
 - 微信支付 (WeChat Pay)
 - 支付宝 (Alipay)
 Below this, there is a note: "To complete payment using Alipay you will be redirected to enter your details and complete the transaction." Buttons for "CARRY ON" and "COMPLETE WITH ALIPAY" are shown at the bottom.
- Conversion Optimization:** Shows a "Experiment Summary" chart titled "Experiments - EU Shipping AB Test". The chart tracks metrics like Add-to-Cart Rate, Session Conversion, and Checkout Conversion over time. A green arrow points from this chart towards the right edge of the slide.

... we make globeal e-commerce easier!



Once upon a time, there was a brave knight...



The evil overlords



© marketoonist.com

The dragon

sw crawl europython2018

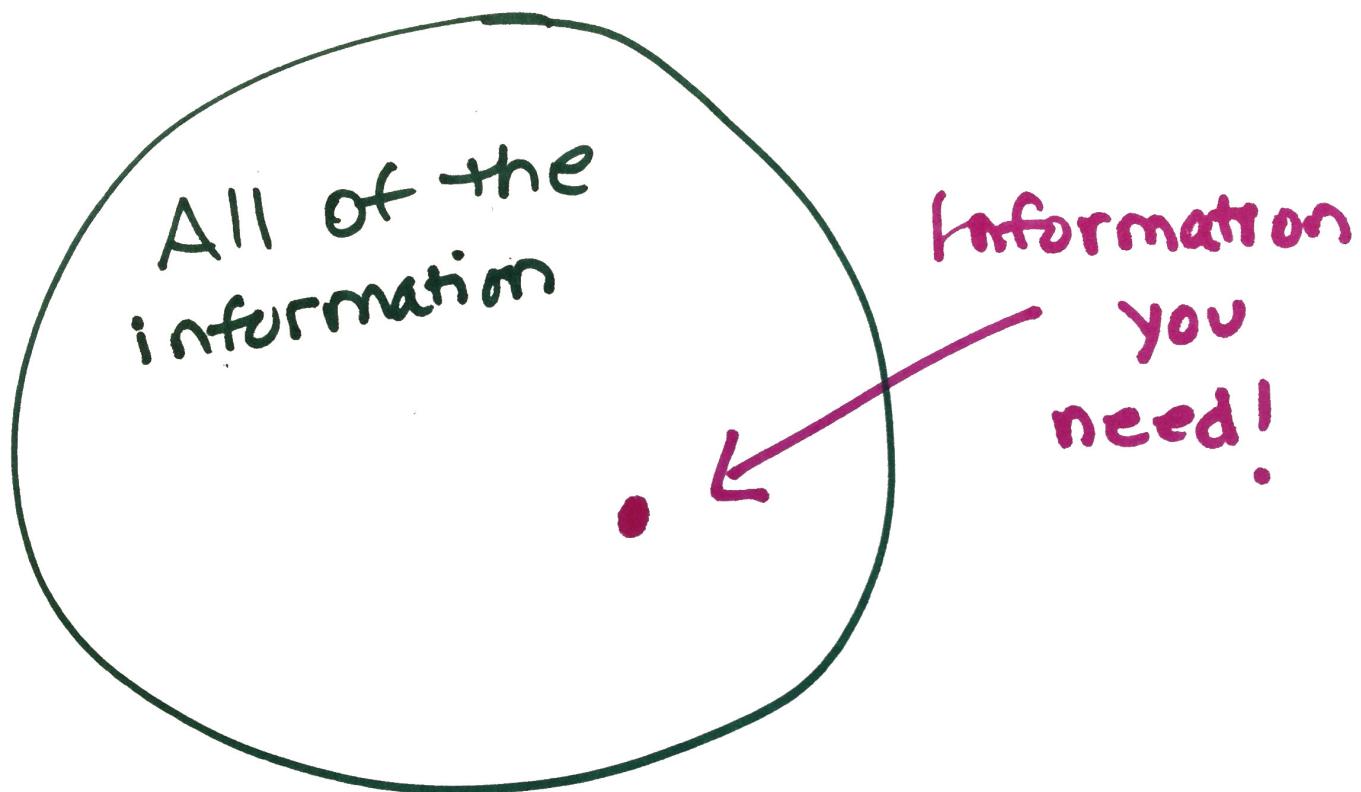


It's just too much...



"After careful consideration of all 437 charts, graphs, and metrics,
I've decided to throw up my hands, hit the liquor store,
and get snockered. Who's with me?!"

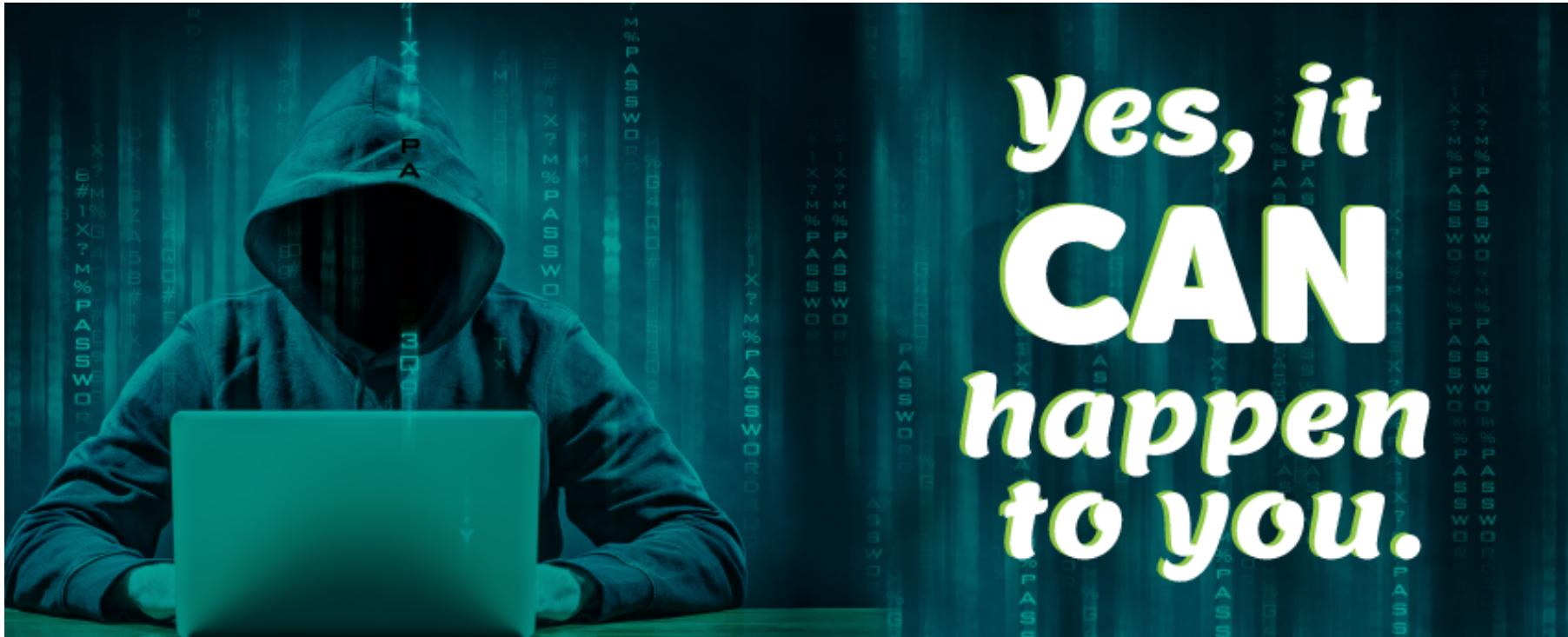
Reality



Problems!

- Understanding the data.
- Training time.
- Hardware requirements (sometimes).
- Overfitting.
- Decreased performance.
- Leaks.
- ...

Sometimes you won't even notice it...



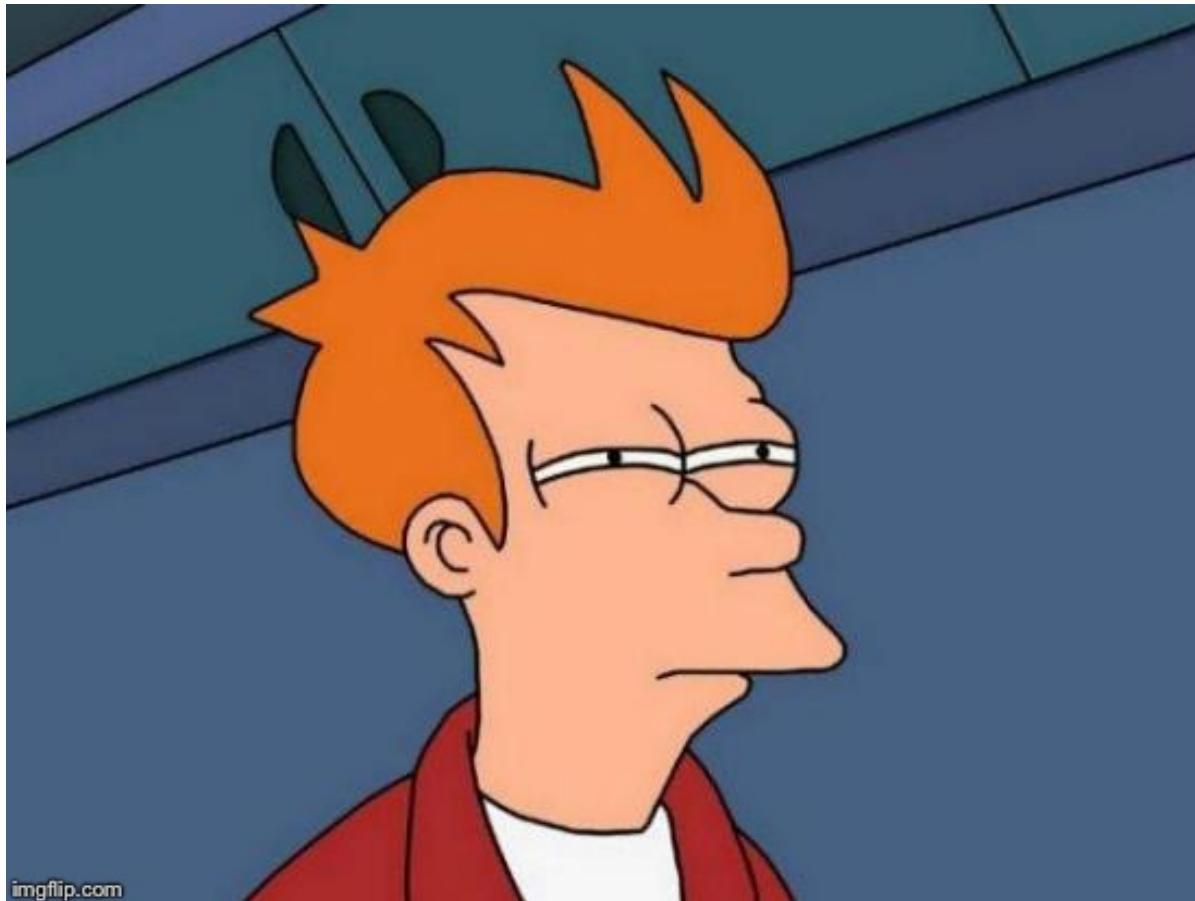
14

When you get 99.9% accuracy 😊



15

... 2.3 seconds later 🤪



16

Let's move on

17

This is just a talk after all...

We have a dataset and a target.

```
df = pd.DataFrame(data['data'], columns=data['feature_names'])  
print(f"Dataset contains {df.shape[0]} rows x {df.shape[1]} columns")
```

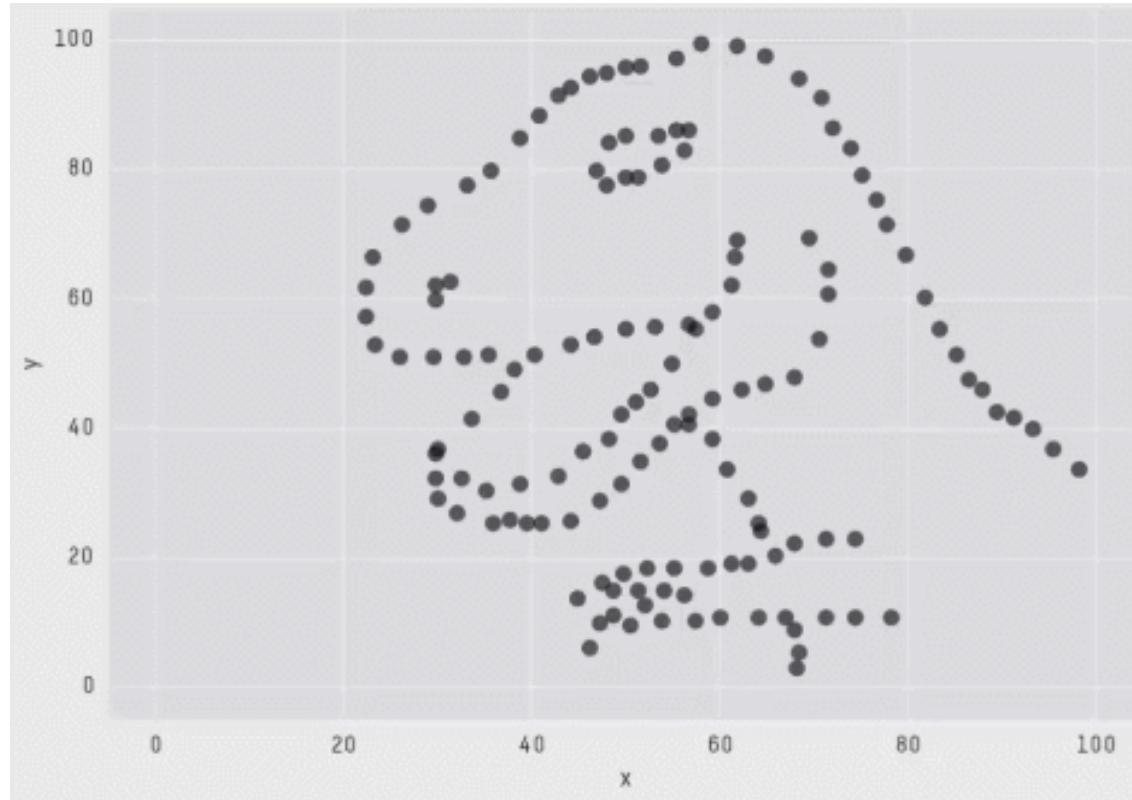
Run

18

Which features do we choose? And how do we choose them?

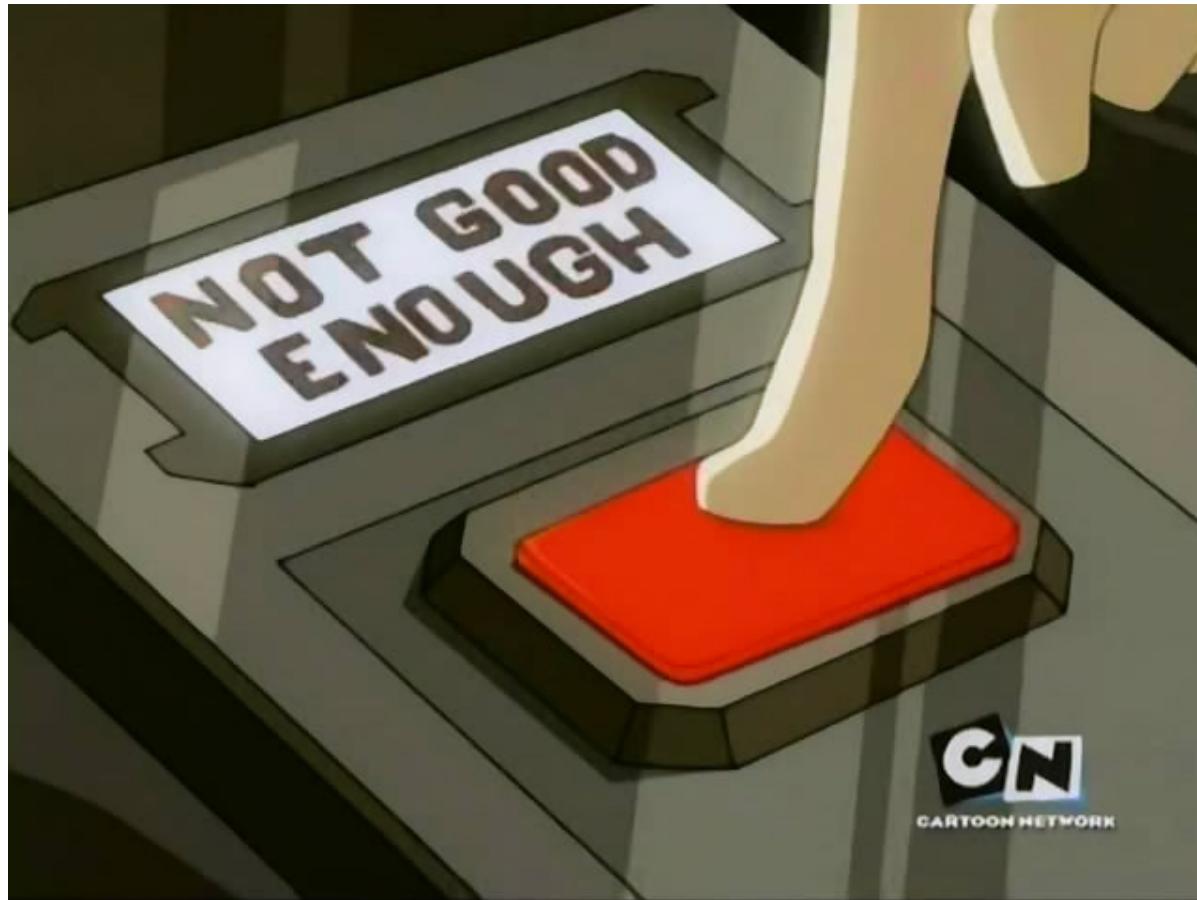
19

Be mindful of your metrics!



X Mean: 54.2659224
Y Mean: 47.8313999
X SD : 16.7649829
Y SD : 26.9342120
Corr. : -0.0642526

Filter methods

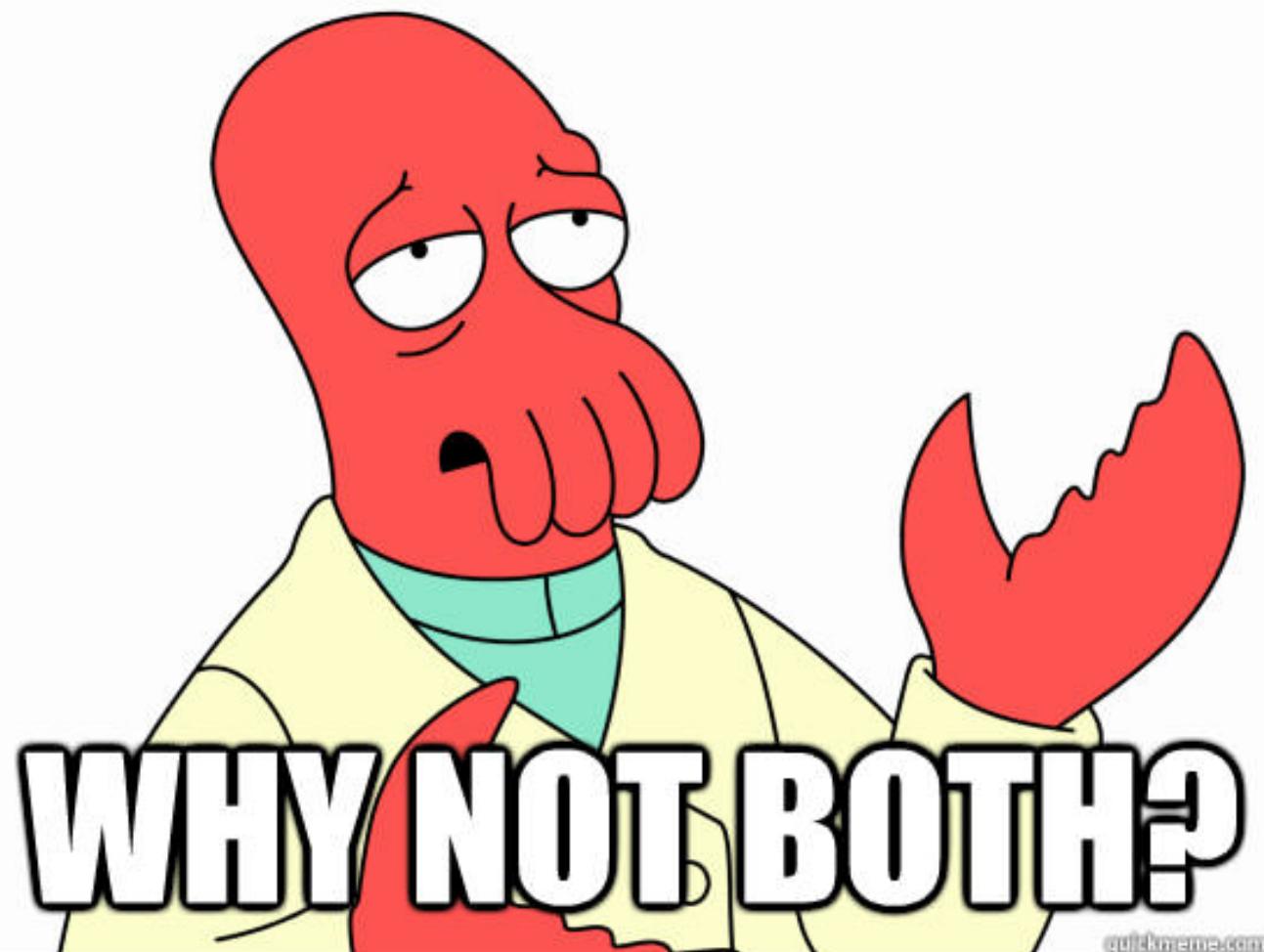


Wrapper Methods



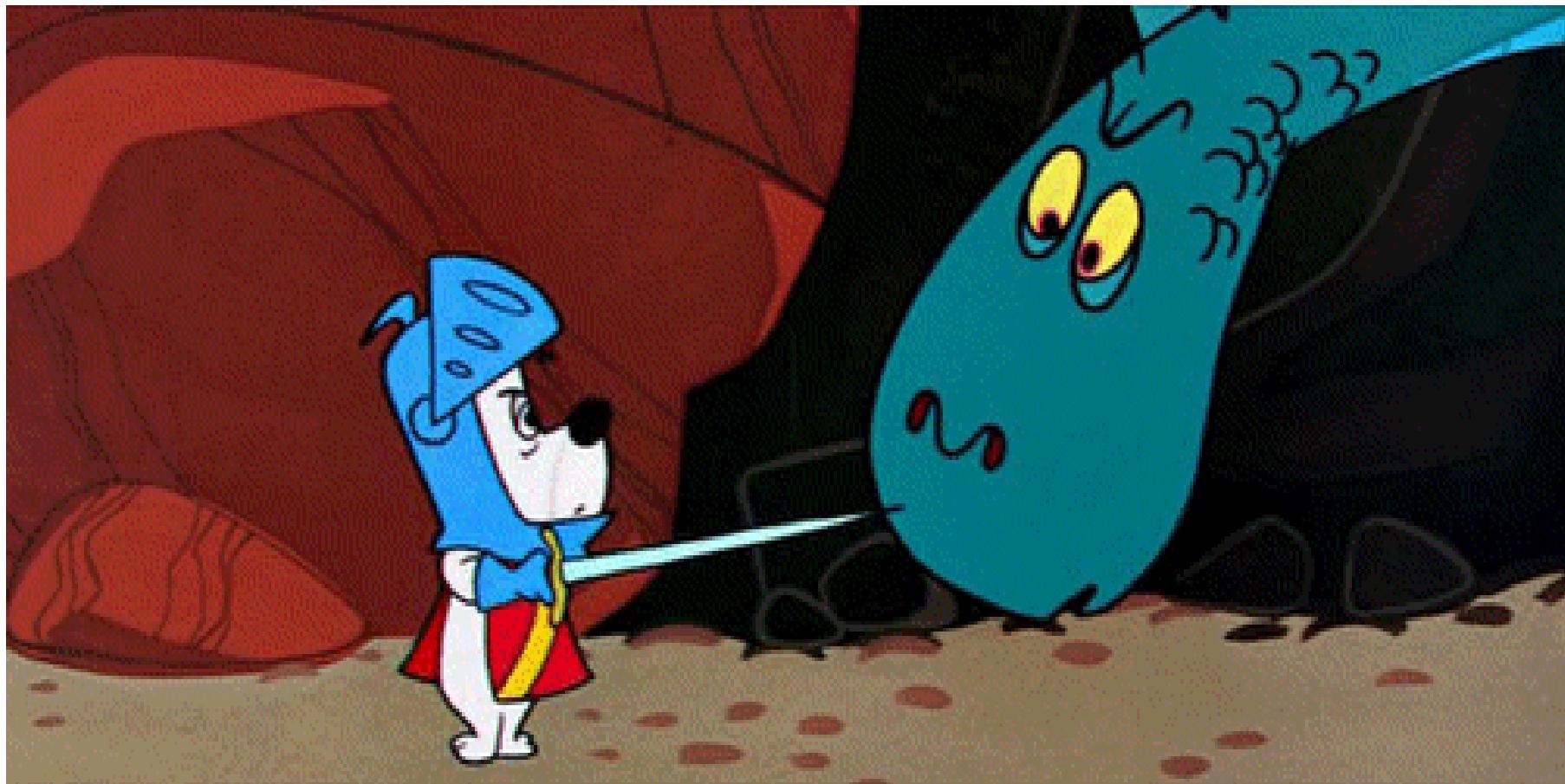
22

Embedded methods



Why are we here again?

24



25

The fight

CAVEAT:

The following code is not complete (to promote clarity).

Errors/edge cases handling/docstrings/comments/... are not included.

DO NOT use the code as is: It will NOT work properly...

The code

We start by importing a bunch of things...

```
import numpy as np
import pandas as pd
from sklearn.base import BaseEstimator

from sklearn.ensemble import RandomForestClassifier, RandomForestRegressor
from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor
from sklearn.feature_selection import VarianceThreshold
```

We set up our class...

```
class EnsembleFeatureSelector(BaseEstimator):  
  
    __regressors = {  
        "RandomForestRegressor": RandomForestRegressor,  
        "DecisionTreeRegressor": DecisionTreeRegressor  
    }  
    __classifiers = {  
        "RandomForestClassifier": RandomForestClassifier,  
        "DecisionTreeClassifier": DecisionTreeClassifier  
    }  
    __others = {  
        "VarianceThreshold": VarianceThreshold,  
    }  
  
    __estimators_by_type = {  
        "classification": ["__classifiers"],  
        "regression": ["__regressors"],  
        "generic_classification": ["__classifiers", "__others"],  
        "generic_regression": ["__regressors", "__others"]  
    }
```

...and we initialise it

```
def __init__(  
    self, number_of_features=-1, analysis_type=None,  
    params=None, min_score=0.7, test_split=0  
):  
  
    ...a bunch of initializations here...
```

Models instantiation

```
def _setup_models(self, params):  
  
    estimators = self.__estimators_by_type.get(self.analysis_type)  
  
    for item in estimators:  
        for label, est in self.__class__.__dict__[  
            f"_{self.__class__.__name__}{item}"  
        ].items():  
            estimator_kwargs = params.get(label, dict())  
            self._active_estimators[label] = est(**estimator_kwargs)
```

Fitting

```
def fit(self, X, y=None):  
  
    number_of_estimators = len(self._active_estimators)  
  
    for n, (key, model) in enumerate(self._active_estimators.items(), start=1):  
        model.fit(X, y)  
  
        self._alive_estimators.add(key)
```

Feature importance by model

```
@staticmethod
def _calculate_importance(model, names=None):
    if hasattr(model, "feature_importances_"):
        return [(n, i) for n, i in enumerate(model.feature_importances_)]
    if hasattr(model, "variances_"):
        return [(n, i) for n, i in enumerate(model.variances_)]
    if hasattr(model, "scores_"):
        return [(n, i) for n, i in enumerate(model.scores_)]
    return list()

def _get_importances(self):
    for key in self._alive_estimators:
        importances = self._calculate_importance(self._active_estimators[key], self.names)

        self._importances[key] = sorted(
            importances, key=operator.itemgetter(1), reverse=True
        )[:self.n_features]
```

Voting

```
def cast_votes(self, min_votes=0):
    if not self._is_fit:
        raise RuntimeError("Ensemble has not been fit on data. Cannot cast votes")

    self._get_importances()

    votes = collections.Counter(
        feature for feature, _ in itertools.chain(*self._importances.values())
    )
    return collections.Counter(
        {feature: vote for feature, vote in votes.items() if vote >= min_votes}
    )
```

Good to go!



34

Let's see it in action!

Remember our dataset?

```
df = pd.DataFrame(data['data'], columns=data['feature_names'])  
print(f"Dataset contains {df.shape[0]} rows x {df.shape[1]} columns")
```

Run

Let's run the selector to see what features are important:

```
EFS = EnsembleFeatureSelector(  
    analysis_type="generic_classification",  
    verbose=1,  
    names=df.columns,  
    number_of_features=5  
)  
  
EFS.fit(df, target)  
votes = EFS.cast_votes()  
  
pretty_print_votes(votes)  
pretty_print_importances(EFS._importances)
```

Run

But does it work?

```
df_train, df_test, target_train, target_test = train_test_split(df, target)

logres.fit(df_train, target_train)
logres_important.fit(df_train[list(votes.keys())], target_train)
```

Run

37

Feature selection:

- simplifies models.
- reduces training time.
- reduces overfitting by enhancing generalization.
- avoids the curse of dimensionality.
- makes data more understandable.
- removes noise.

Things should be as simple as possible, but not simpler...

A. Einstein

38

Q/A



Thank you

Tags: [flow](#), [machinelearning](#), [datascience](#), [python](#), [2019](#) (#ZgotmplZ)

Pietro Mascolo

Director of Machine Learning - Flow Commerce

@iz4we (<http://twitter.com/iz4we>)

Slides and code: bit.ly/2MPPg5j (<https://bit.ly/2MPPg5j>)

