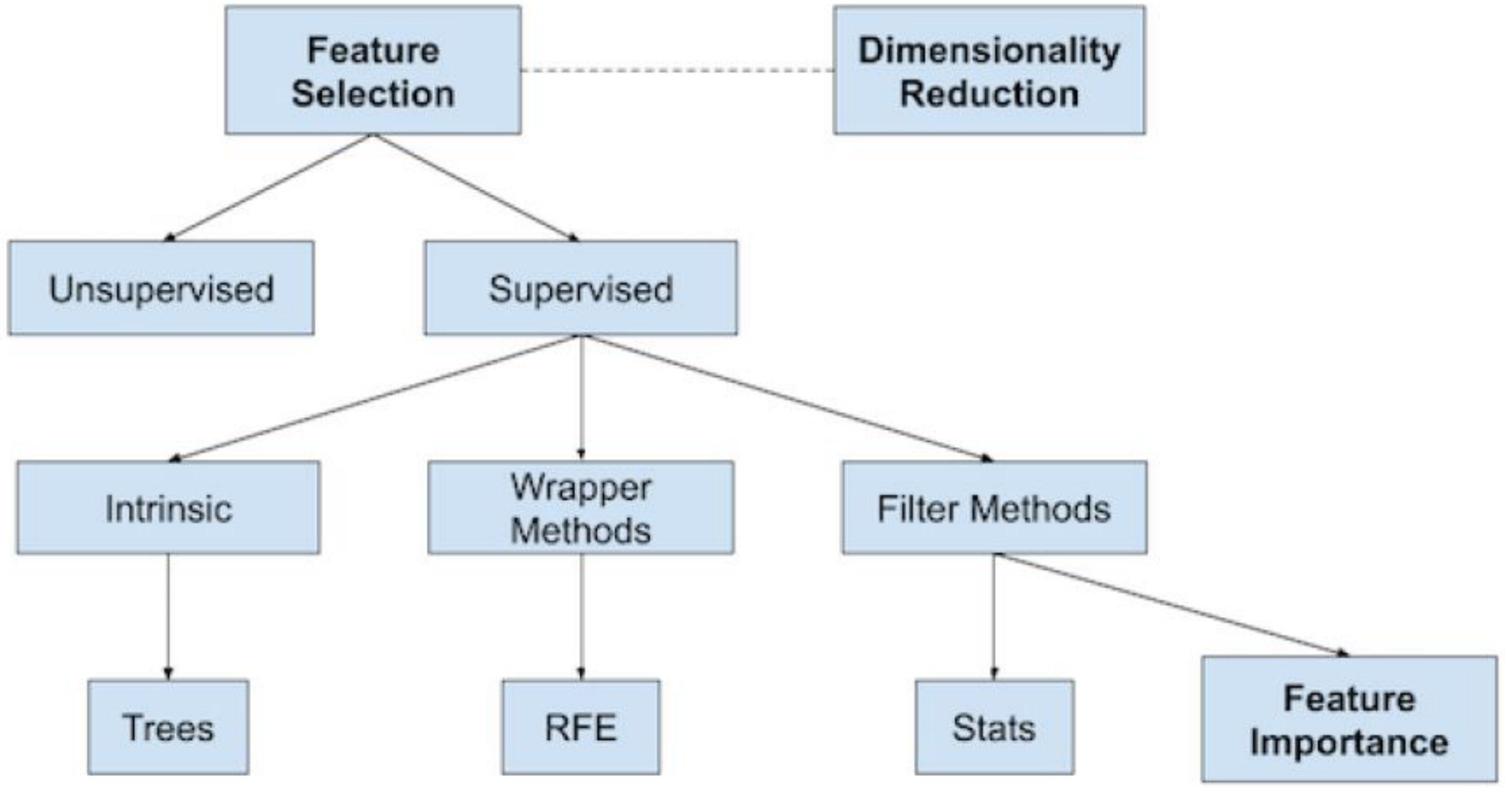
Lesson #05
Feature Selection

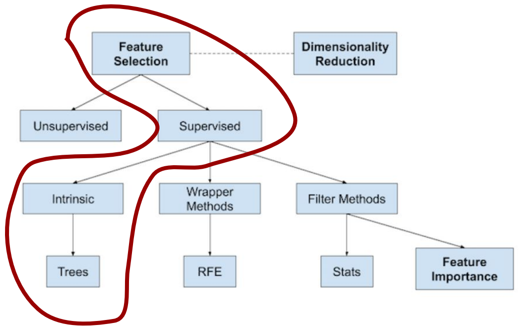
Table of Contents

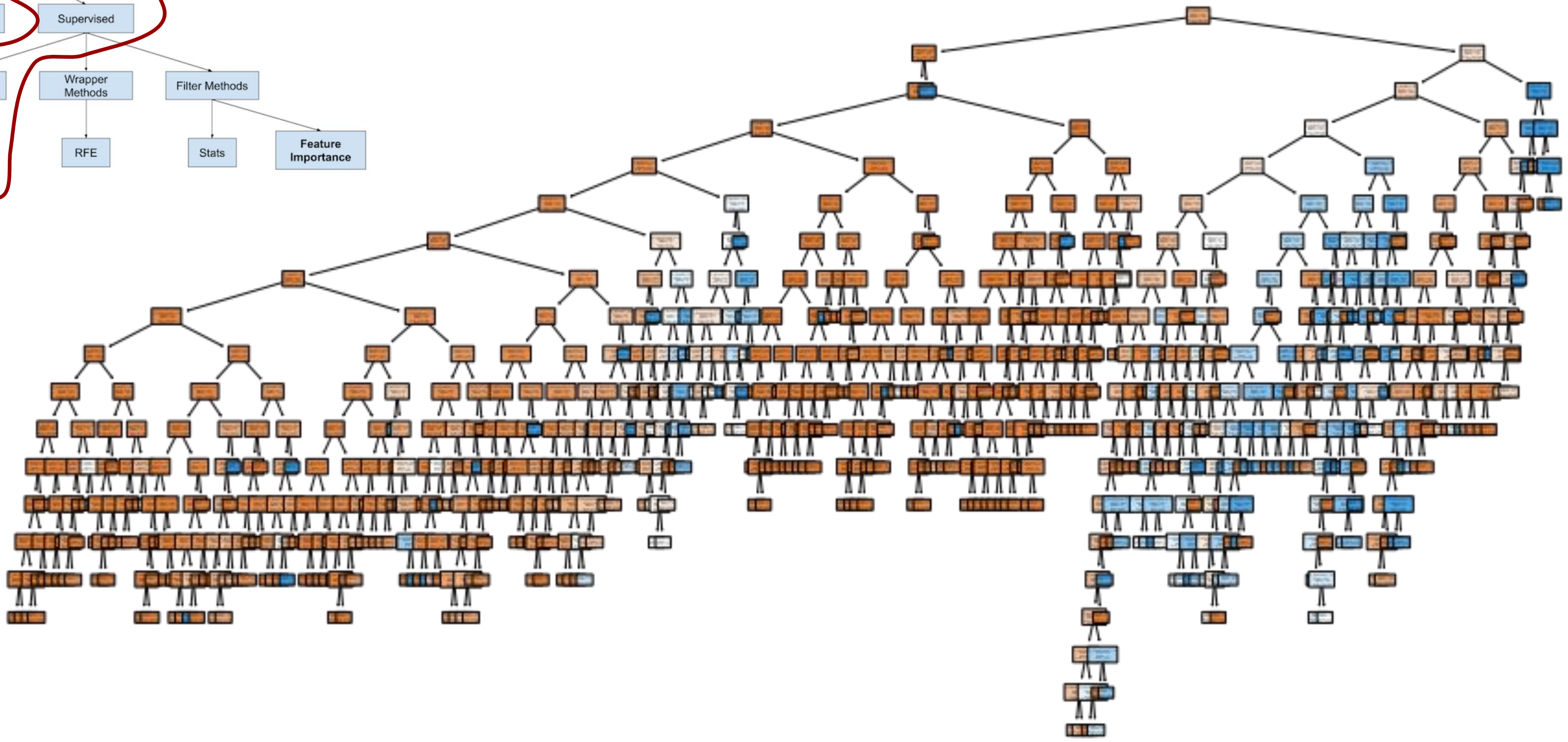
- Feature selection for ML
- Hypothesis test
  - Significant test
  - Chi-squared test
- Pipelines & ML Viz

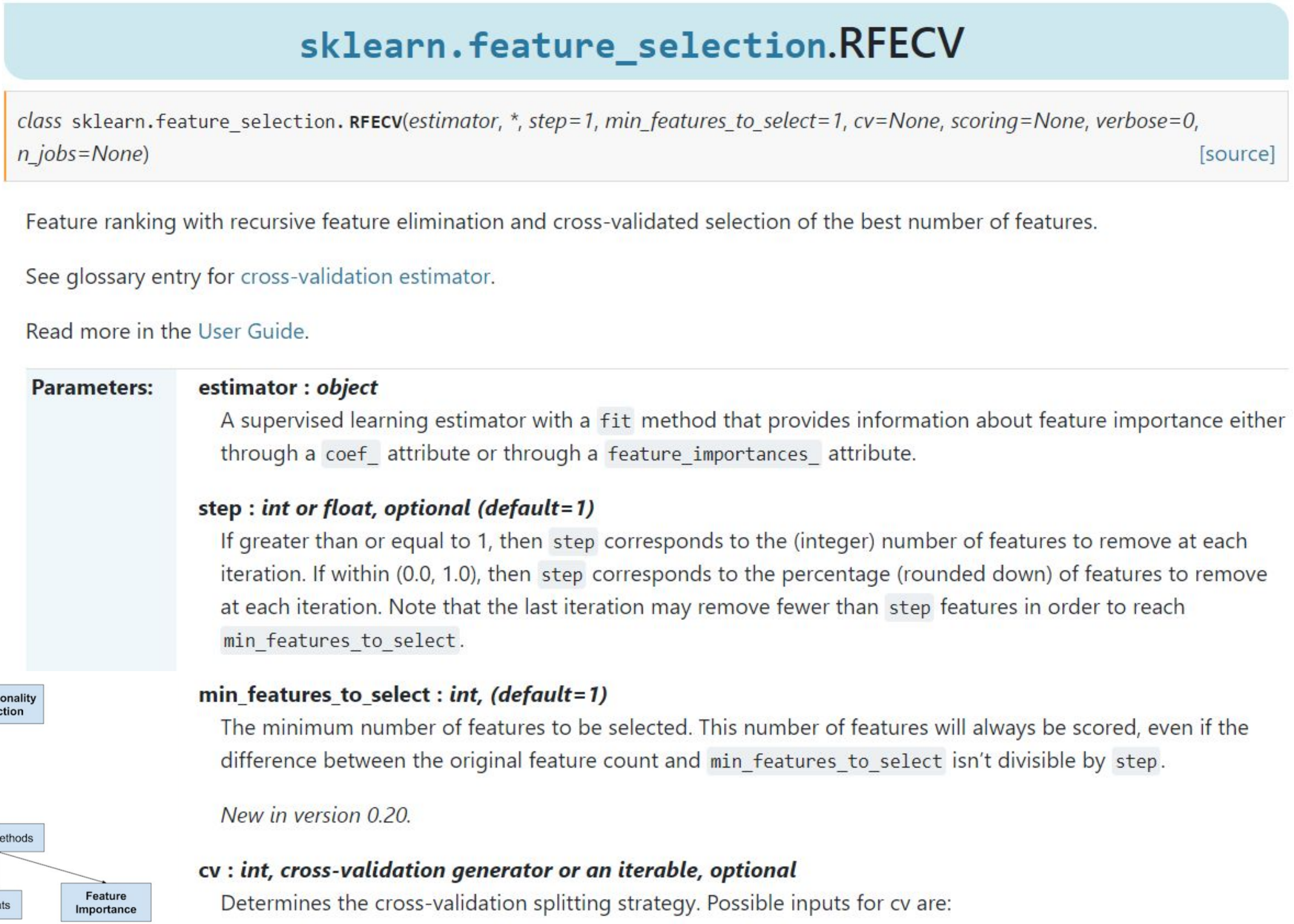
A common problem in applied machine learning is determining whether **input features are relevant to the outcome to be predicted**

Feature selection


Feature Selection
Dimensionality Reduction
Unsupervised
Supervised
Intrinsic
Wrapper Methods
Filter Methods
Trees
RFE
Stats
Feature Importance


Feature Selection
Dimensionality Reduction
Unsupervised
Supervised
Intrinsic
Wrapper Methods
Filter Methods
Trees
RFE
Stats
Feature Importance





sklearn.feature_selection.RFECV

class `sklearn.feature_selection.`**`RFECV`**(*estimator, *, step=1, min_features_to_select=1, cv=None, scoring=None, verbose=0, n_jobs=None*) [source]

Feature ranking with recursive feature elimination and cross-validated selection of the best number of features.

See glossary entry for cross-validation estimator.

Read more in the User Guide.

Parameters:

estimator : *object*

A supervised learning estimator with a `fit` method that provides information about feature importance either through a `coef_` attribute or through a `feature_importances_` attribute.

step : *int or float, optional (default=1)*

If greater than or equal to 1, then `step` corresponds to the (integer) number of features to remove at each iteration. If within (0.0, 1.0), then `step` corresponds to the percentage (rounded down) of features to remove at each iteration. Note that the last iteration may remove fewer than `step` features in order to reach `min_features_to_select`.

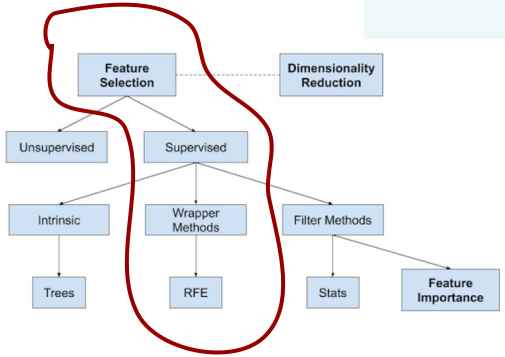
min_features_to_select : *int, (default=1)*

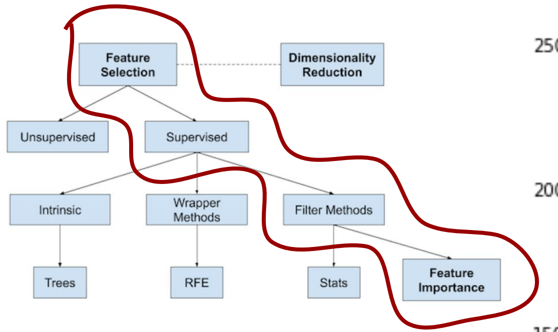
The minimum number of features to be selected. This number of features will always be scored, even if the difference between the original feature count and `min_features_to_select` isn't divisible by `step`.

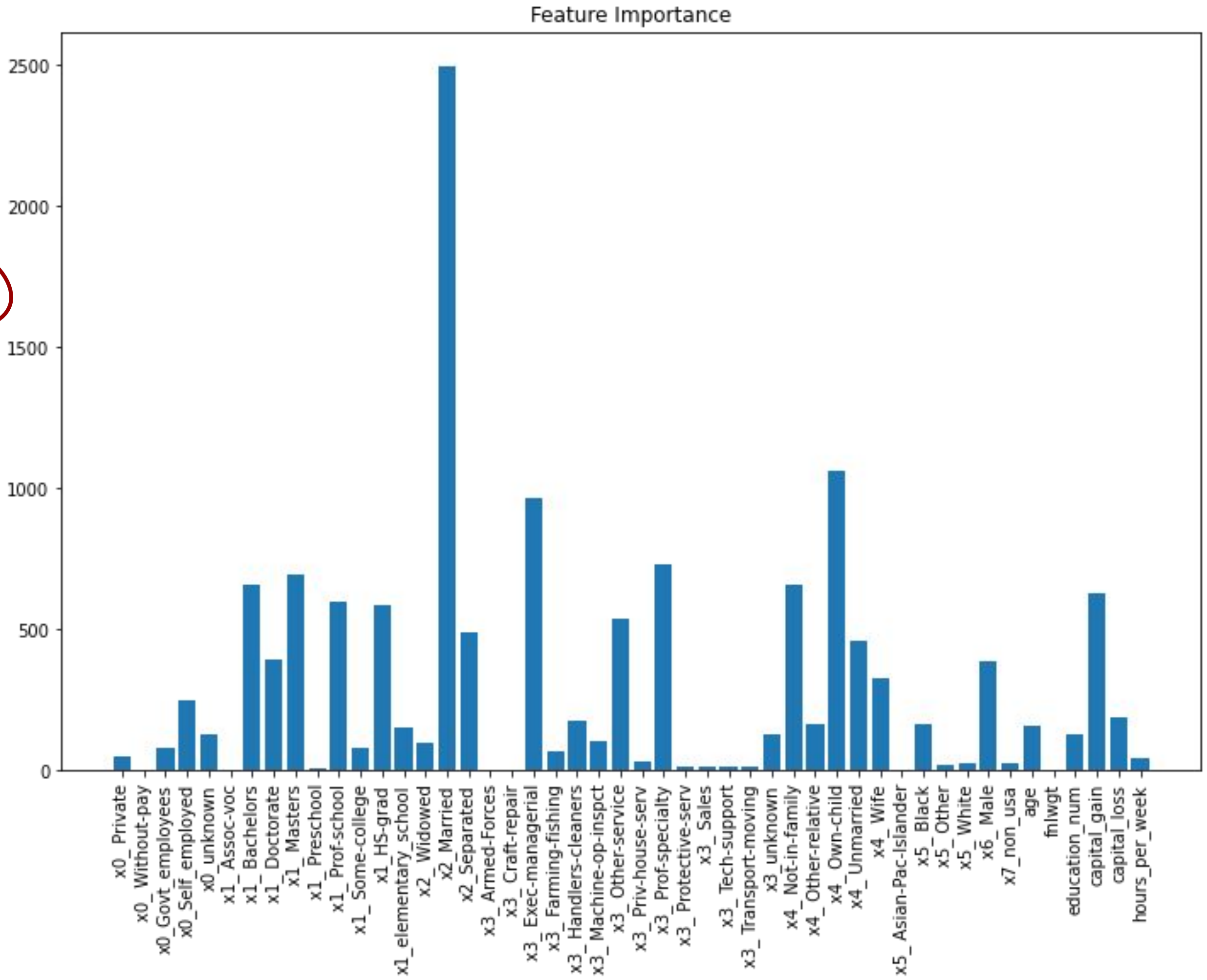
New in version 0.20.

cv : *int, cross-validation generator or an iterable, optional*

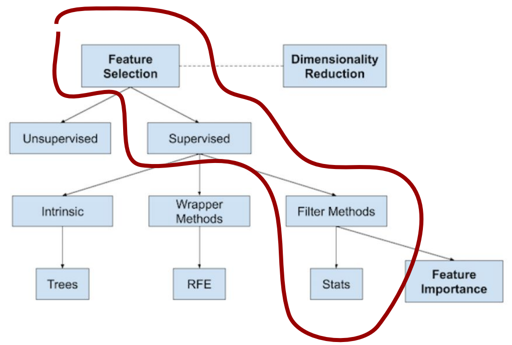
Determines the cross-validation splitting strategy. Possible inputs for cv are:

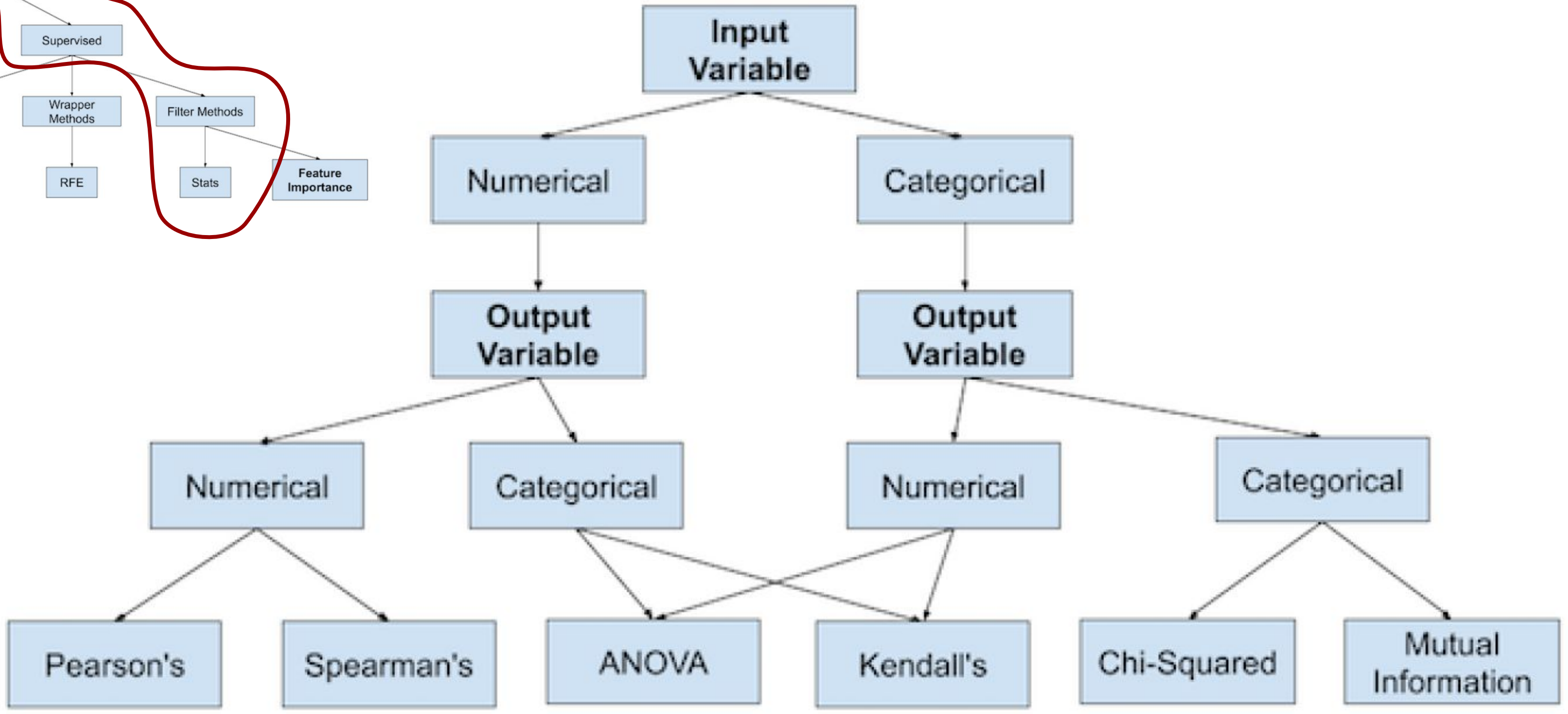


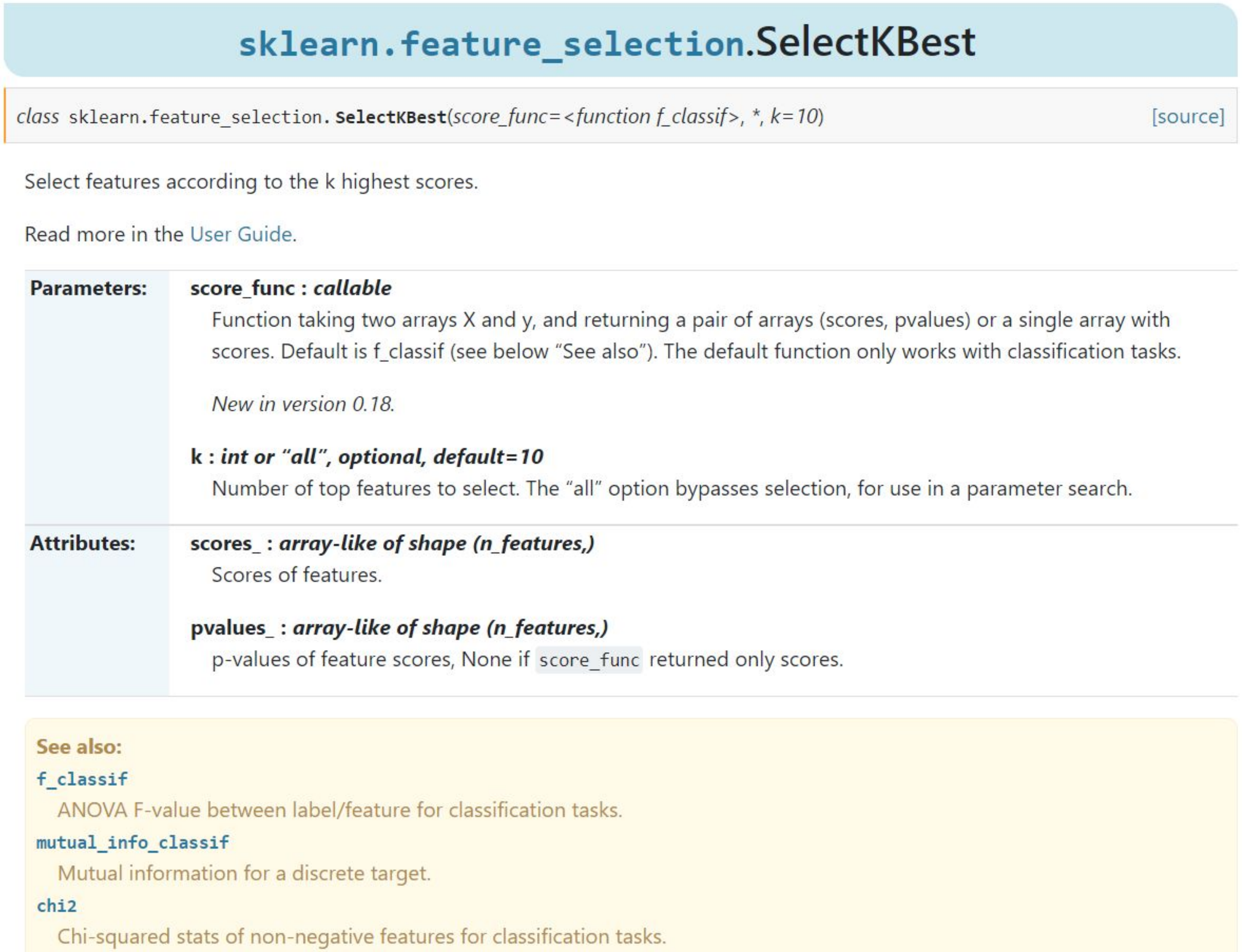




An agnostic method is to score input features using a model and use a filter-based feature selection method


Feature Selection
Dimensionality Reduction
Unsupervised
Supervised
Intrinsic
Wrapper Methods
Filter Methods
Trees
RFE
Stats
Feature Importance


Input Variable
Numerical
Categorical
Output Variable
Output Variable
Numerical
Categorical
Numerical
Categorical
Pearson's
Spearman's
ANOVA
Kendall's
Chi-Squared
Mutual Information



sklearn.feature_selection.SelectKBest

class `sklearn.feature_selection.SelectKBest`(*score_func=<function f_classif>, *, k=10*) [source]

Select features according to the k highest scores.

Read more in the User Guide.

Parameters:	**score_func : *callable*** Function taking two arrays X and y, and returning a pair of arrays (scores, pvalues) or a single array with scores. Default is f_classif (see below "See also"). The default function only works with classification tasks. *New in version 0.18.* **k : *int or "all", optional, default=10*** Number of top features to select. The "all" option bypasses selection, for use in a parameter search.
Attributes:	**scores_ : *array-like of shape (n_features,)*** Scores of features. **pvalues_ : *array-like of shape (n_features,)*** p-values of feature scores, None if `score_func` returned only scores.

See also:

`f_classif`
ANOVA F-value between label/feature for classification tasks.

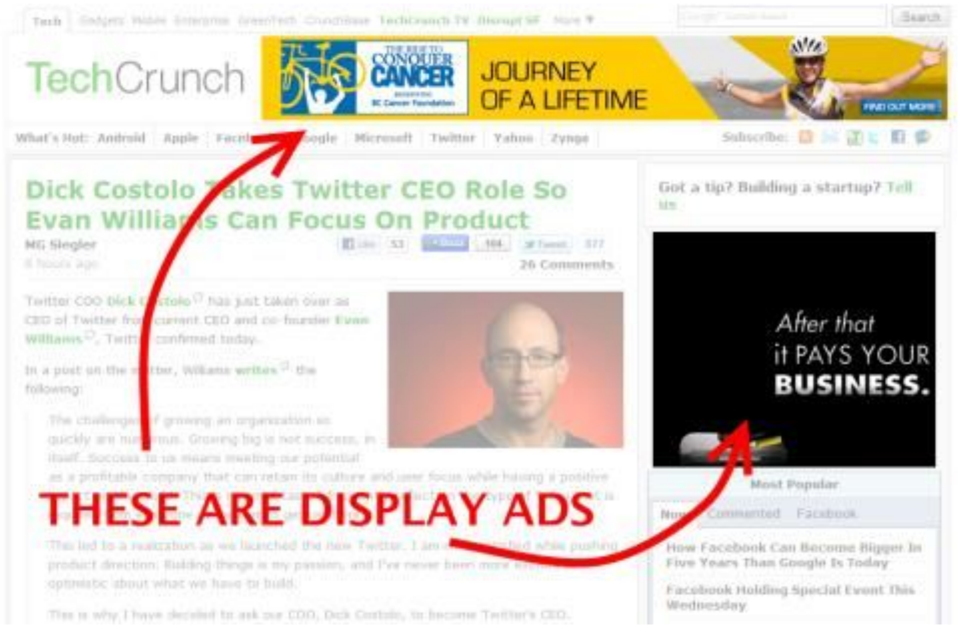
`mutual_info_classif`
Mutual information for a discrete target.

`chi2`
Chi-squared stats of non-negative features for classification tasks.

WHAT IS A HYPOTHESIS?

"A hypothesis is an idea that can be tested"

Did raising the price of a product cause a meaningful drop in sales?



Dick Costolo Takes Twitter CEO Role So Evan Williams Can Focus On Product

MG Siegler

Has a new banner ad on a website caused a meaningful drop in the user engagement?

Null vs. Alternative Hypothesis

Null Hypothesis

H_0

A statement about a population parameter.

We test the likelihood of this statement being true in order to decide whether to accept or reject our alternative hypothesis.

Can include =, ≤, or ≥ sign.

Alternative Hypothesis

H_a

A statement that directly contradicts the null hypothesis.

We determine whether or not to accept or reject this statement based on the likelihood of the null (opposite) hypothesis being true.

Can include a ≠, >, or < sign.

ThoughtCo.

365√DataScience

$H_0 : \mu_0 > \$125{,}000$

$H_1 : \mu_0 \leq \$125{,}000$

THE NULL HYPOTHESIS IS THE STATEMENT WE ARE TRYING TO REJECT. THEREFORE THE NULL IS THE PRESENT STATE OF AFFAIRS WHILE THE ALTERNTIVE IS OUR PERSONAL OPINION.

EXAMPLE

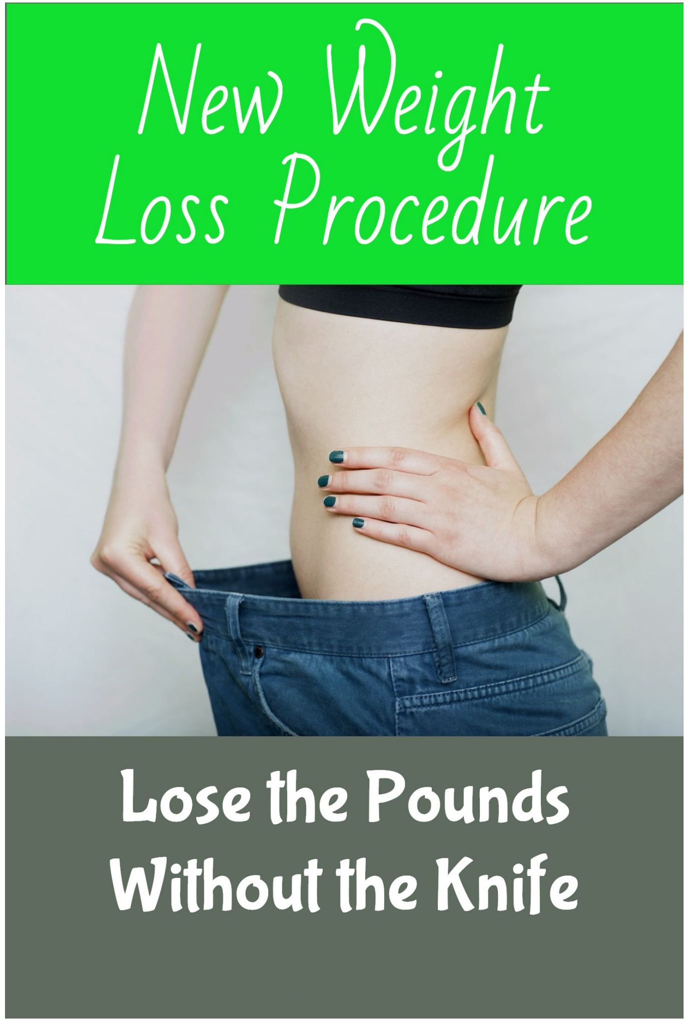
$H_0: \mu_0 = \$125{,}000$

Accept if: $\bar{x}$ is close enough to the true mean

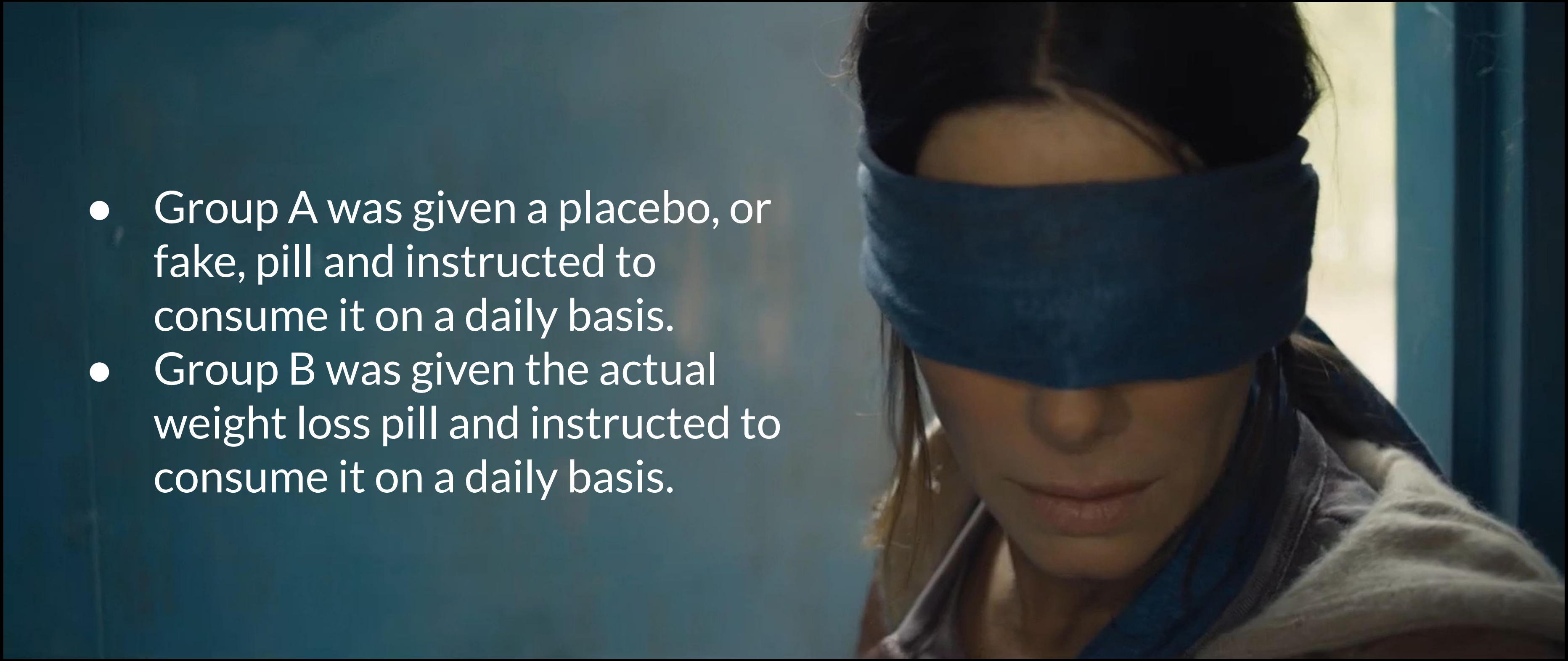
Reject if: $\bar{x}$ is too far from the true mean

A new weight loss pill helped people lose more weight:

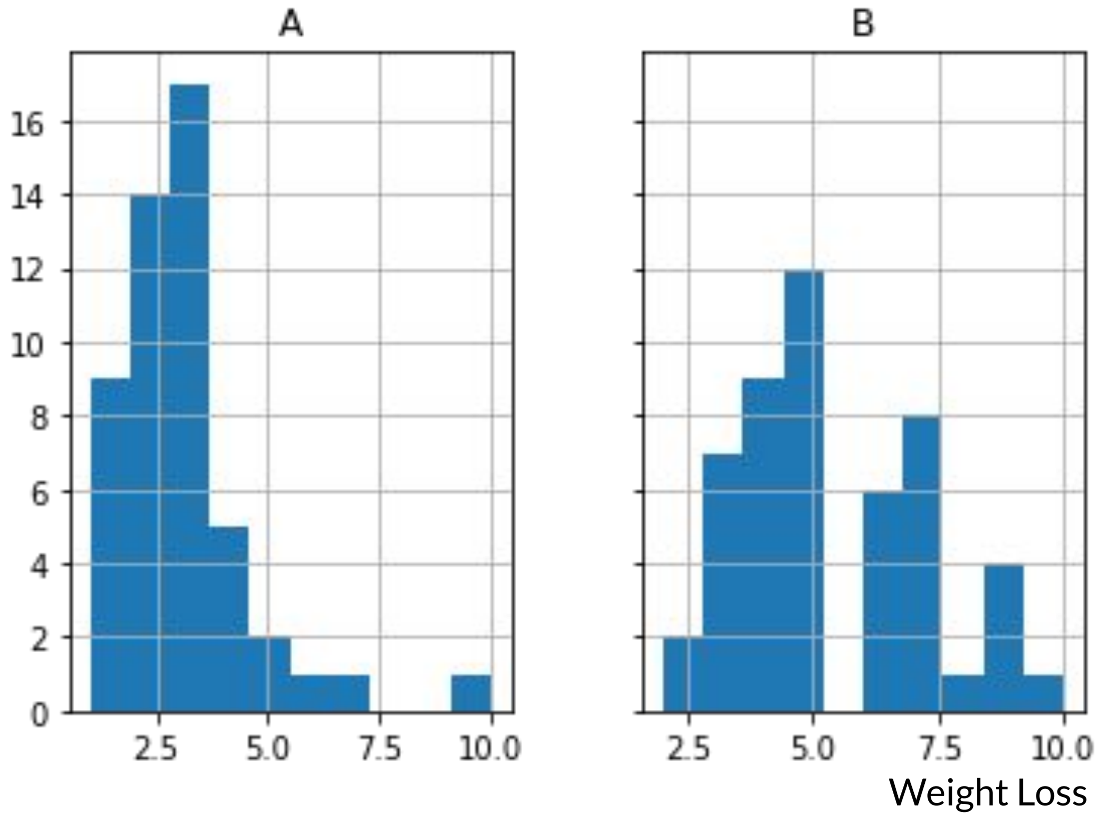
H_0: patients who went on the weight loss pill lost no more weight than those who didn't.
H_1: patients who went on the weight loss pill lost more weight than those who didn't



Blind Experiment



- Group A was given a placebo, or fake, pill and instructed to consume it on a daily basis.
- Group B was given the actual weight loss pill and instructed to consume it on a daily basis.



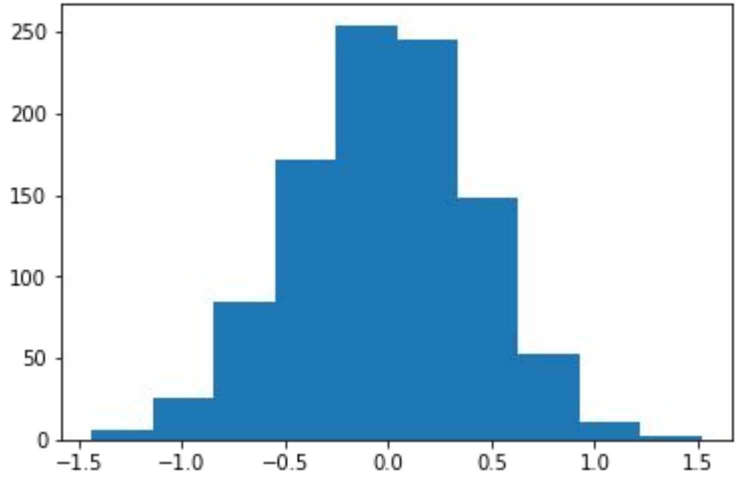
Null hypothesis: $\bar{x}_b - \bar{x}_a = 0$

Alternative hypothesis: $\bar{x}_b - \bar{x}_a > 0$

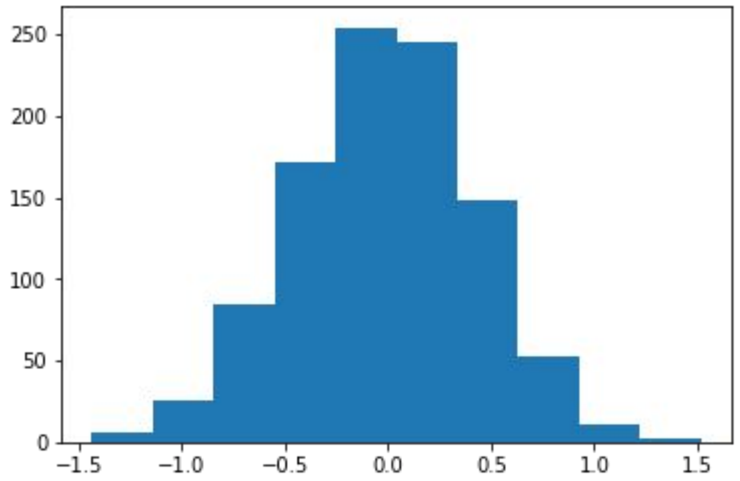
$\bar{x}_b - \bar{x}_a = 2.52$

Permutation Test

```
mean_difference = 2.52
mean_differences = []
for i in range(1000):
    group_a = []
    group_b = []
    for value in all_values:
        assignment_chance = np.random.rand()
        if assignment_chance >= 0.5:
            group_a.append(value)
        else:
            group_b.append(value)
    iteration_mean_difference = np.mean(group_b) - np.mean(group_a)
    mean_differences.append(iteration_mean_difference)
plt.hist(mean_differences)
```



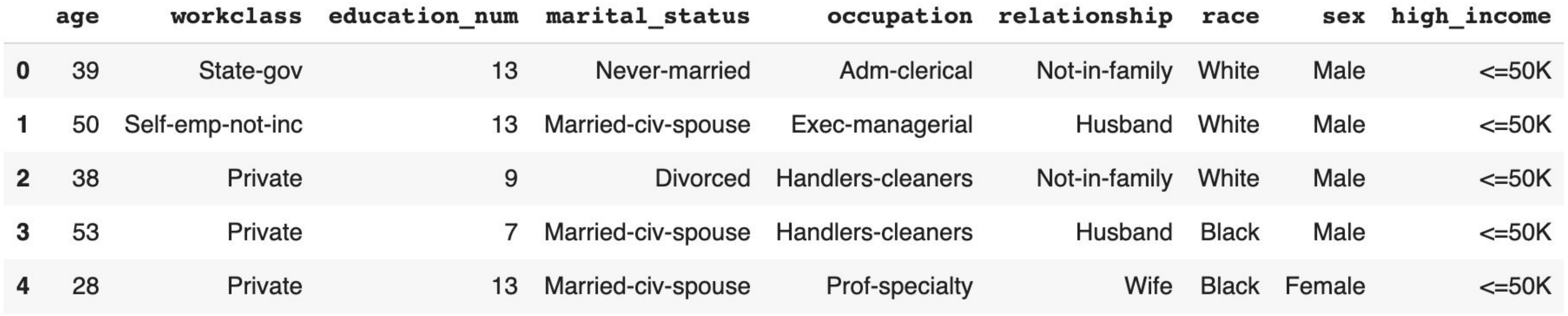
```
[(0.036814725890355504, 8),
 (-0.16000000000000014, 7),
 (-0.370935367322360З, 6),
 (-0.447115384615З846, 6),
 (-0.046474358974359475, 6),
 (0.17021276595744705, 6),
 (0.16326530612244916, 6),
 (0.08992372541148086, 5),
 (0.18840579710144967, 5),
 (-0.0305098354074671, 5)]
```



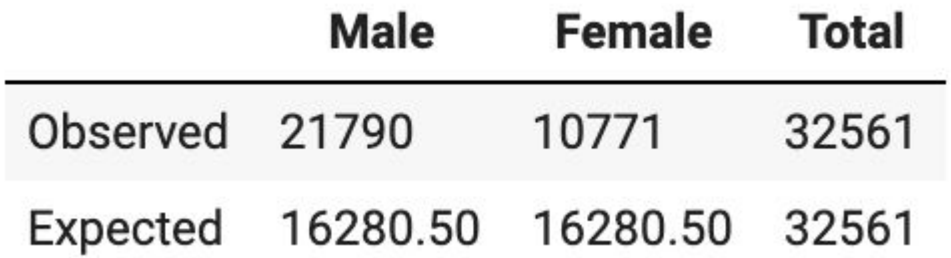
```python
frequencies = []
for sp in sampling_distribution.keys():
    if sp >= 2.52:
        frequencies.append(sampling_distribution[sp])
p_value = np.sum(frequencies) / 1000
p_value    0.0
```

In general, it's good practice to set the **p value** threshold before conducting the study:

- if the **p value** is less than the threshold, we:
  - **reject the null hypothesis** that there's no difference in mean amount of weight lost by participants in both groups,
  - **accept the alternative hypothesis** that the people who consumed the weight loss pill lost more weight,
  - conclude that the weight loss pill does affect the amount of weight people lost.
- if the **p value** is greater than the threshold, we:
  - **accept the null hypothesis** that there's no difference in the mean amount of weight lost by participants in both groups,
  - **reject the alternative hypothesis** that the people who consumed the weight loss pill lost more weight,
  - conclude that the weight loss pill doesn't seem to be effective in helping people lose more weight.



	age	workclass	education_num	marital_status	occupation	relationship	race	sex	high_income
0	39	State-gov	13	Never-married	Adm-clerical	Not-in-family	White	Male	<=50K
1	50	Self-emp-not-inc	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	<=50K
2	38	Private	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	<=50K
3	53	Private	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	<=50K
4	28	Private	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	<=50K



	Male	Female	Total
Observed	21790	10771	32561
Expected	16280.50	16280.50	32561

We don't have any way to determine if there's a statistically significant difference between the two groups, and if we need to investigate further

Chi-Squared Test

we need a way to figure out what the chi-squared value represents

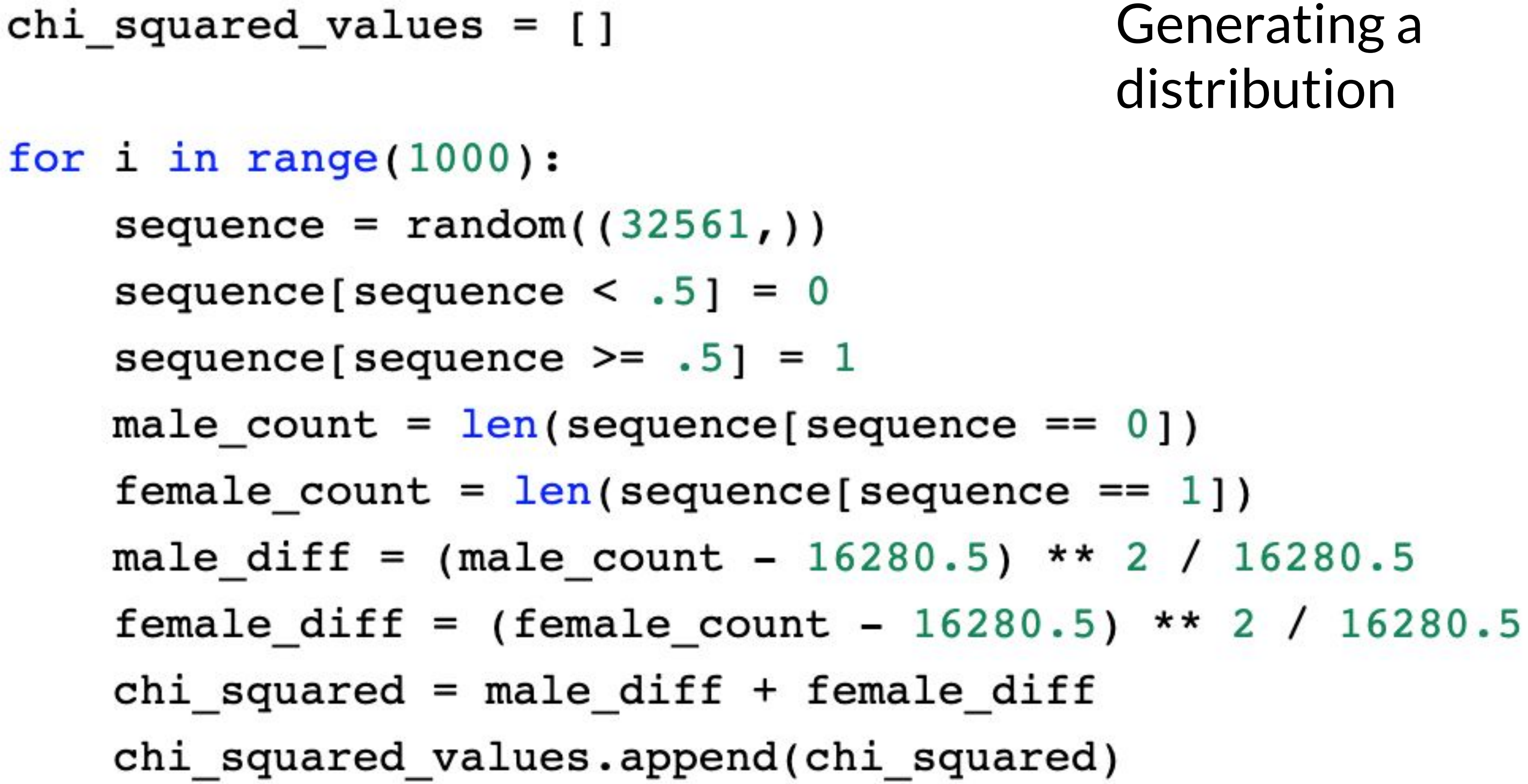
Chi-Squared Value

$$\chi^2 = \sum_{i=1}^{k} \frac{(observed - expected)^2}{expected}$$

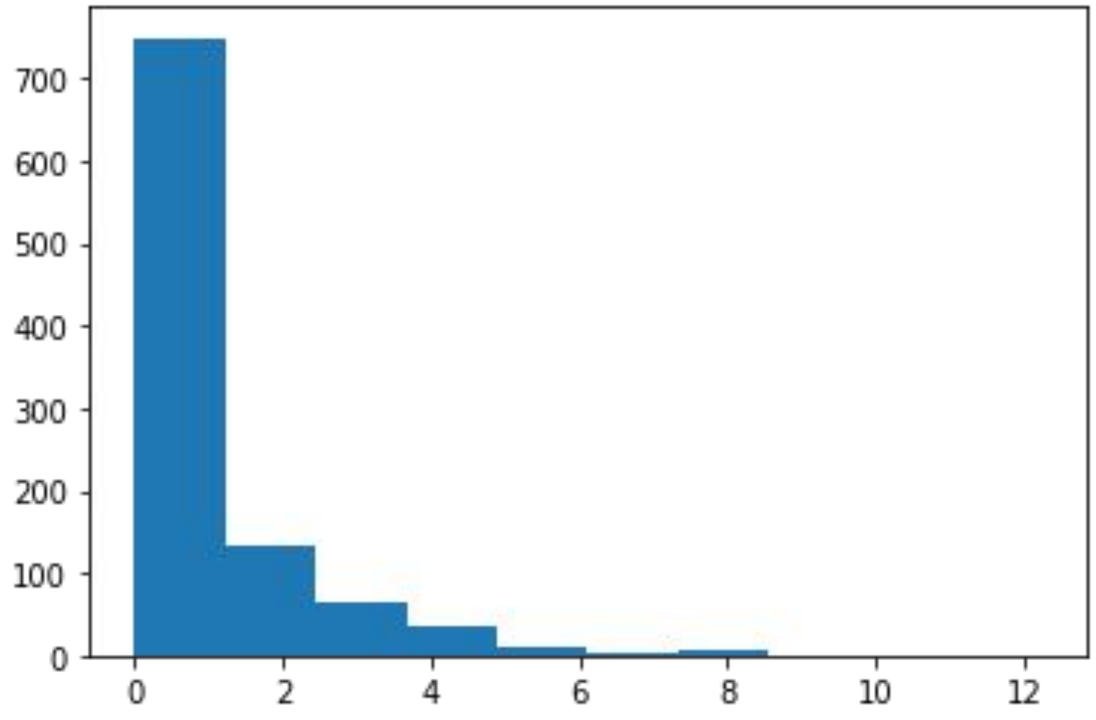
```
female_diff = (data.sex.value_counts()[1] - data.shape[0]* 0.5)**2/(data.shape[0]*0.5)
male_diff = (data.sex.value_counts()[0] - data.shape[0]* 0.5)**2/(data.shape[0]*0.5)
chi2_census = female_diff + male_diff
chi2_census
```

```
3728.950615767329
```

Generating a distribution



```
chi_squared_values = []

for i in range(1000):
     sequence = random((32561,))
     sequence[sequence < .5] = 0
     sequence[sequence >= .5] = 1
     male_count = len(sequence[sequence == 0])
     female_count = len(sequence[sequence == 1])
     male_diff = (male_count - 16280.5) ** 2 / 16280.5
     female_diff = (female_count - 16280.5) ** 2 / 16280.5
     chi_squared = male_diff + female_diff
     chi_squared_values.append(chi_squared)
```



Chi-squared values

How many values are greater than 3728.95 (our chi-squared value)?

p-value = 0

This would indicate that we need to investigate our data collection techniques more closely to figure out why such a result occurred.

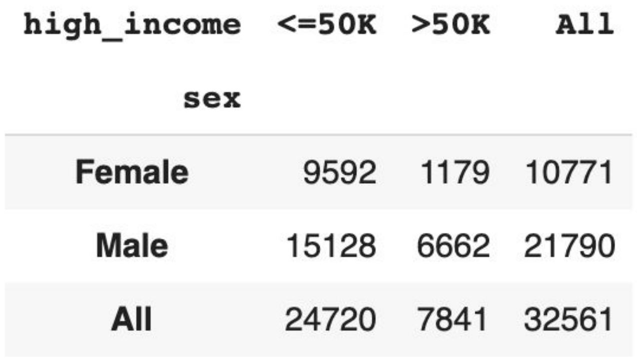
	Male	Female	Total
Observed	21790	10771	32561
Expected	16280.50	16280.50	32561

```
import numpy as np
from scipy.stats import chisquare

observed = np.array([21790, 10771])
expected = np.array([16280.50,16280.50])
chisquare_value, pvalue = chisquare(observed, expected)

3728.950615767329, 0
```

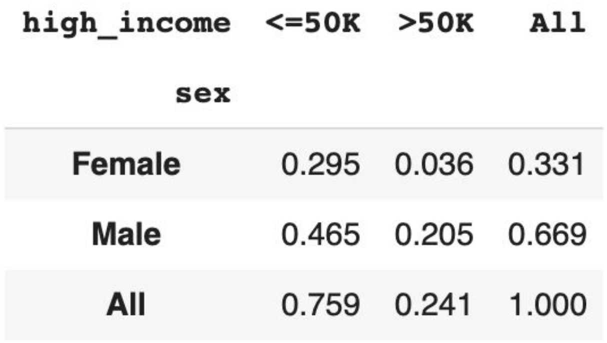
How two categorical columns interact?

Chi-Square statistic will test whether there is a significant difference (dependent vs independent) in the observed vs the expected frequencies of both variables.

```
pd.crosstab(data["sex"], [data["high_income"]],margins=True,normalize=True)
```



high_income	<=50K	>50K	All
sex			
Female	9592	1179	10771
Male	15128	6662	21790
All	24720	7841	32561



high_income	<=50K	>50K	All
sex			
Female	0.295	0.036	0.331
Male	0.465	0.205	0.669
All	0.759	0.241	1.000

- The Null hypothesis is that there is NO association between both variables.
- The Alternate hypothesis says there is evidence to suggest there is an association between the two variables.

If we reject the null hypothesis, it's an important variable (dependent) to use in your model.

To reject the null hypothesis, the calculated P-Value needs to be below a defined threshold.

Expected values

```
males_over50k = .669 * .241 * 32561
males_under50k = .669 * .759 * 32561
females_over50k = .331 * .241 * 32561
females_under50k = .331 * .759 * 32561
```

high_income	<=50K	>50K	All
sex			
Female	0.295	0.036	0.331
Male	0.465	0.205	0.669
All	0.759	0.241	1.000

```
observed = np.array([9592, 1179, 15128,     6662])
expected = np.array([8180.27, 2597.42, 16533.53, 5249.78])

chisq_value, pvalue_gender_income = chisquare(observed, expected)

    1517.59531656486, 0
```

Be Pythonic

```
chisq_value, pvalue_gender_race, df, expected = chi2_contingency(pd.crosstab(data["sex"],
                                                                         [data["high_income"]]))
```

The function return:

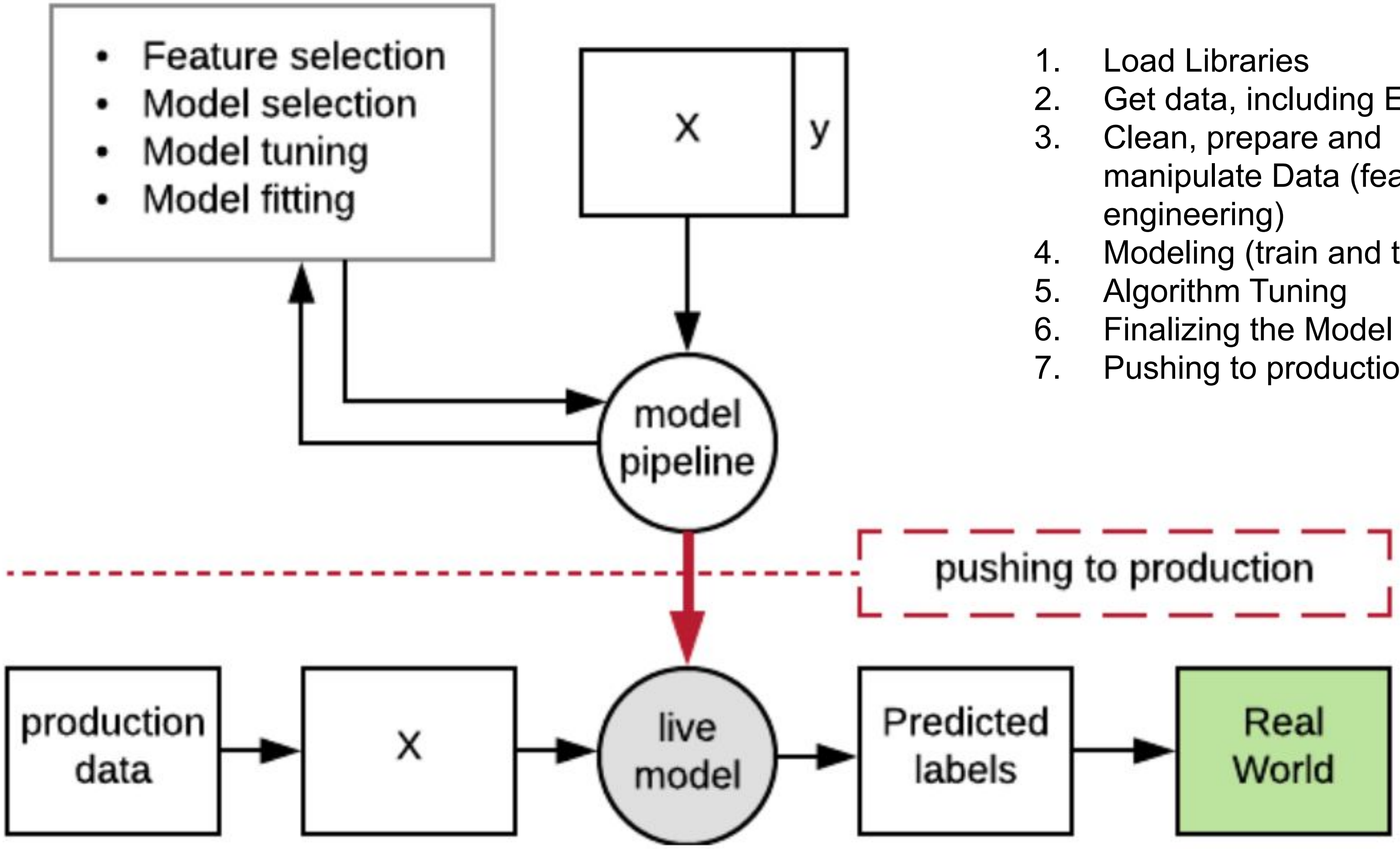
- Chi-squared value
- P-value
- Degree of freedom
- Expected value

Feature selection

```python
# all chi-squared values
chi_dict = {}
# only categorical columns are used in chi-squared test
columns = income.select_dtypes("object").columns.to_list()[:-1]

# eliminate the high_income column
for name in columns:
  chisq_value, pvalue_all, df, expected = chi2_contingency(pd.crosstab(income[name],[income["high_income"]]))
  chi_dict[name] = (chisq_value, pvalue_all)

sorted(chi_dict.items(), key=lambda kv: kv[1][0],reverse=True)
```

```
[('relationship', (6699.07689685885, 0.0)),
 ('marital_status', (6517.741653663022, 0.0)),
 ('education', (4429.653302288619, 0.0)),
 ('occupation', (4031.974280247181, 0.0)),
 ('sex', (1517.813409134445, 0.0)),
 ('workclass', (1045.7085997281692, 2.026505431120716e-220)),
 ('race', (330.9204310085741, 2.305960610160958e-70)),
 ('native_country', (317.2303857833171, 2.2113858852543023e-44))]
```

Everything ends in **Pipelines**



1. Load Libraries
2. Get data, including EDA
3. Clean, prepare and manipulate Data (feature engineering)
4. Modeling (train and test)
5. Algorithm Tuning
6. Finalizing the Model
7. Pushing to production

Yellowbrick

latest

Search docs

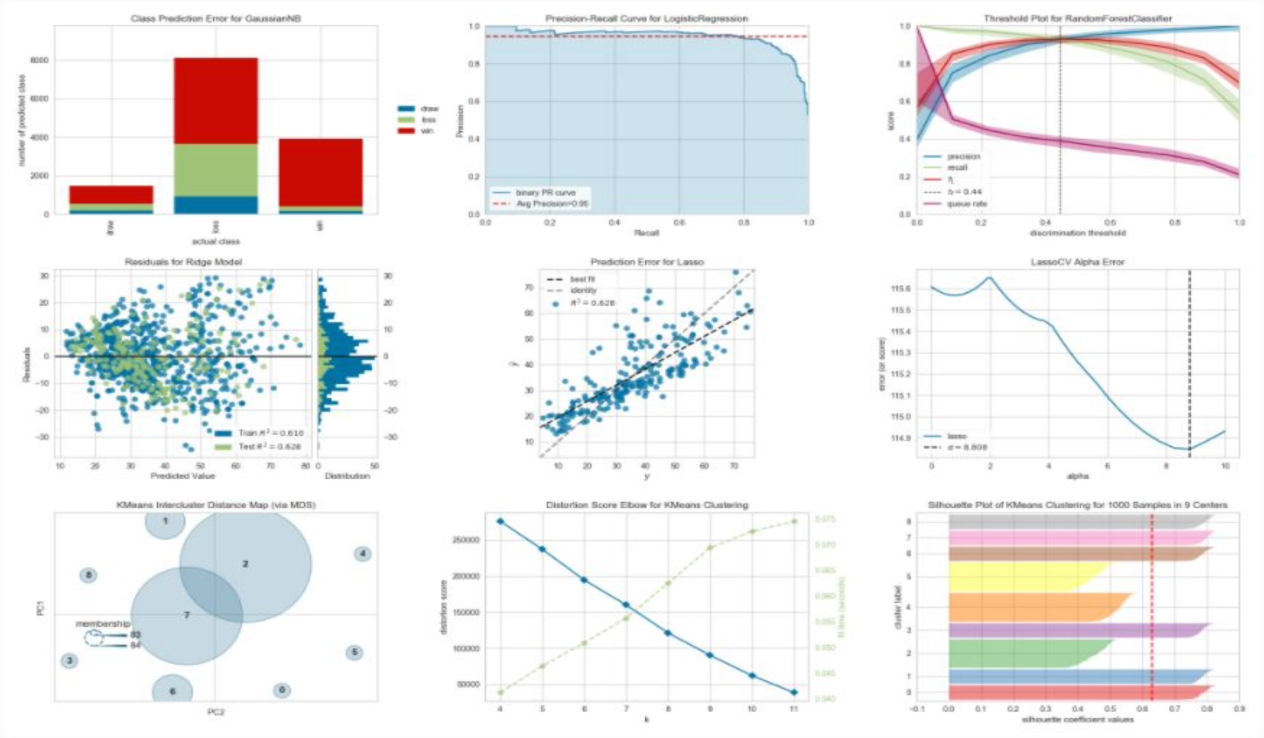
- Quick Start
- Model Selection Tutorial
- Visualizers and API
- Oneliners
- Contributing
- Effective Matplotlib
- Yellowbrick for Teachers
- Gallery
- About
- Frequently Asked Questions
- User Testing Instructions
- Code of Conduct
- Changelog
- Governance

Read the Docs v: latest

Docs » Yellowbrick: Machine Learning Visualization

Edit on GitHub

Yellowbrick: Machine Learning Visualization



Yellowbrick extends the Scikit-Learn API to make model selection and hyperparameter tuning easier. Under the hood, it's using Matplotlib.

Hands on ....

Lesson #05 - Feature Selection.ipynb
Lesson #05 - End to End Pipeline.ipynb