Removing Features with Low Variance

- A simple baseline approach to feature selection.
- It removes all features whose variance doesn’t meet some threshold.
- At the very least, we should remove all zero-variance features, i.e. features that have the same value in all samples.

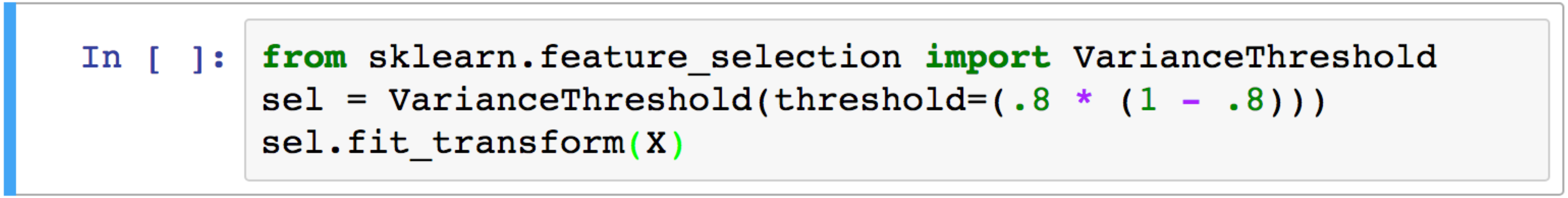
Removing Features with Low Variance

- A simple baseline approach to feature selection.
- It removes all features whose variance doesn’t meet some threshold.
- At the very least, we should remove all zero-variance features, i.e. features that have the same value in all samples.
- As an example, suppose that we have a dataset with boolean features, and we want to remove all features that are either one or zero (on or off) in more than 80% of the samples.
- Boolean features are Bernoulli random variables, and the variance of such variables is given by

$$Var[X] = p(1-p)$$

Removing Features with Low Variance

In sklearn, you may input the desired level of variance for VarianceThreshold selector.

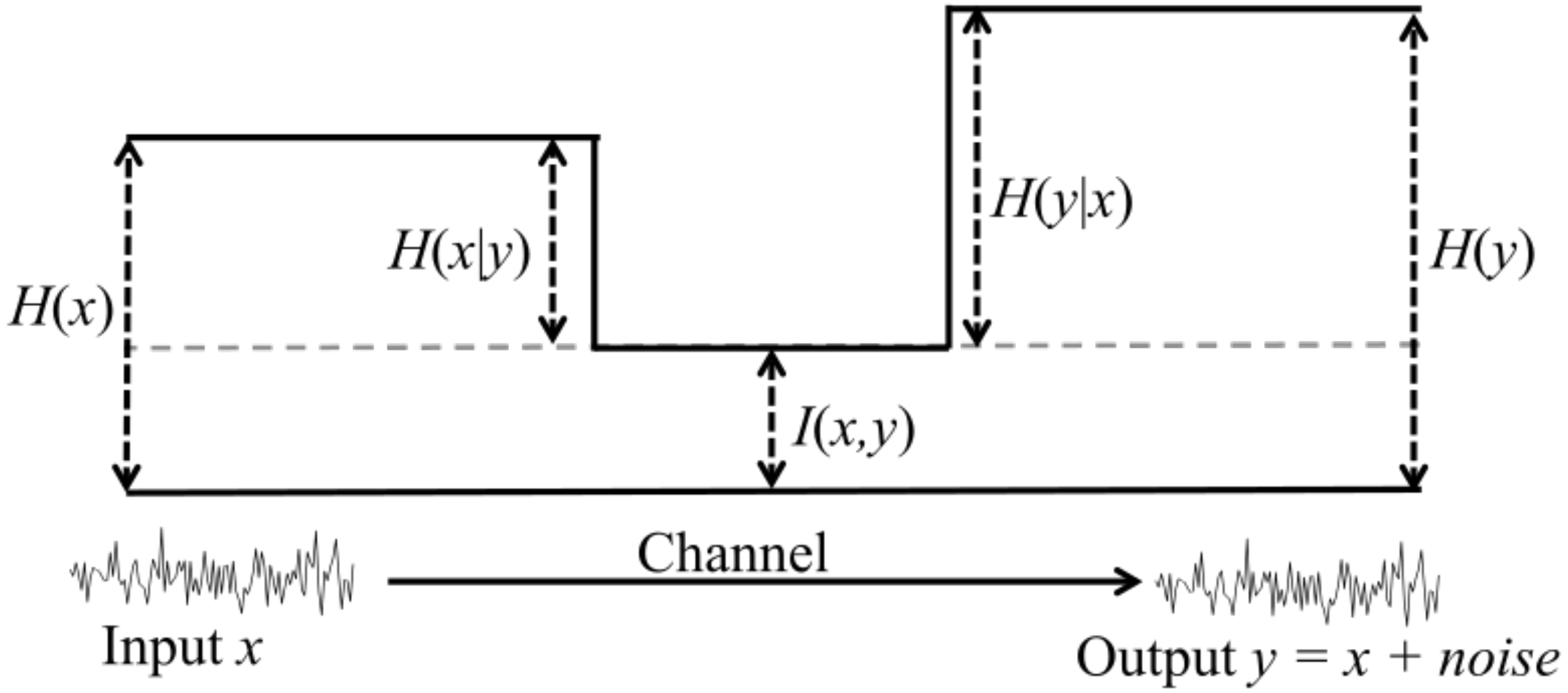


```
from sklearn.feature_selection import VarianceThreshold
sel = VarianceThreshold(threshold=(.8 * (1 - .8)))
sel.fit_transform(X)
```

- sklearn will automatically drop the columns where variance does not meet the desired level.
- Note that this can take care of one-hot encoded features with small number of ones.

Mutual Information

- Mutual Information I(X;Y) is the same as information gain, it is the information variable X and Y share or relevancy between two variables.
- Mutual information is symmetric, I(X;Y) is the same as I(Y;X).

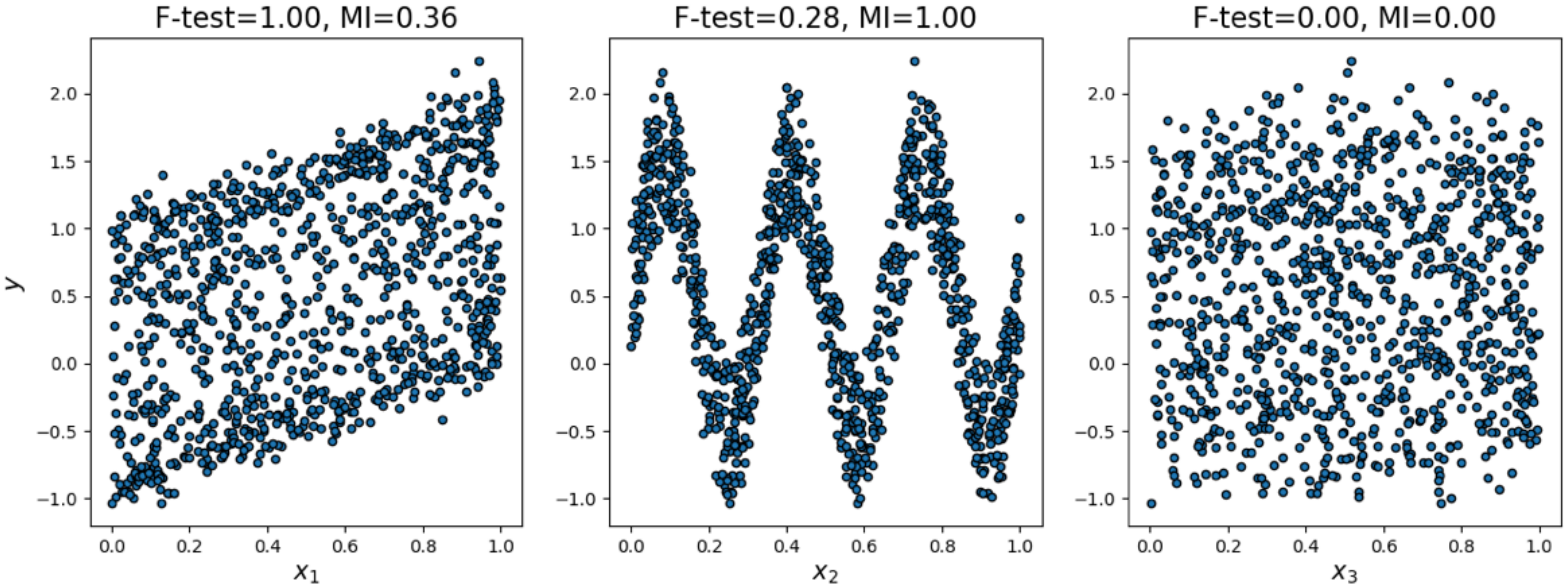


$$
\begin{aligned}
I(X;Y) &\equiv \mathrm{H}(X) - \mathrm{H}(X|Y) \\
&\equiv \mathrm{H}(Y) - \mathrm{H}(Y|X) \\
&\equiv \mathrm{H}(X) + \mathrm{H}(Y) - \mathrm{H}(X,Y) \\
&\equiv \mathrm{H}(X,Y) - \mathrm{H}(X|Y) - \mathrm{H}(Y|X)
\end{aligned}
$$

$$H(X,Y) = -\sum_{x}\sum_{y} P(x,y)\log_2[P(x,y)]$$

F-Test and Mutual Information

- As F-test captures only linear dependency, it rates x1 as the most discriminative feature.
- On the other hand, mutual information can capture any kind of dependency between variables and it rates x2 as the most discriminative feature, which probably agrees better with our intuitive perception for this example.
- Both methods correctly marks x3 as irrelevant.



Mutual Information

- Mutual information methods can capture any kind of statistical dependency, but being nonparametric, they require more samples for accurate estimation.
- Mutual information is a non-negative value, the higher the more information shared between two variables.
- Can be calculated for both regression and classification problem.
  - sklearn mutual_info_regression: http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.mutual_info_regression.html
  - sklearn mutual_info_classif: http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.mutual_info_classif.html

Mutual Information

- For mutual information, you might need to identify each feature as discrete or continuous before proceeding.
- Unlike F-Test, mutual information does not have any p-value, so you may not know whether a particular value of MI is significant or not. They are used as feature ranking measure, not thresholded measure.

X^2 Test of Independence

- For two categorical variables with r classes and c classes respectively, you can make a contingency table that has r rows and c columns.
- The chi square test can be thought of as a test of independence. In a test of independence the null and alternative hypotheses are:
    - Ho: The two categorical variables are independent.
    - Ha: The two categorical variables are related.

http://math.hws.edu/javamath/ryan/ChiSquare.html

X^2 Test of Independence

- Example: we want to know whether continents (X) impact the frequency of different types of malaria (Y).

	Asia	Africa	South America	**Total**
Malaria A	31	14	45	90
Malaria B	2	5	53	60
Malaria C	53	45	2	100
Totals	86	64	100	250

X^2 Test of Independence

- We can use the equation to compute X^2

$$\chi_c^2 = \sum_i \frac{(FO_i - FE_i)^2}{FE_i}$$

	Asia	Africa	South America	**Total**
Malaria A	31	14	45	**90**
Malaria B	2	5	53	**60**
Malaria C	53	45	2	**100**
Totals	**86**	**64**	**100**	**250**

(90/250)*(86/250)*250

Observed	Expected	|O -E|	$(O - E)^2$	$(O - E)^2 / E$
31	30.96	0.04	0.0016	0.0000516
14	23.04	9.04	81.72	3.546
45	36.00	9.00	81	2.25
2	20.64	18.64	347.45	16.83
5	15.36	10.36	107.33	6.99
53	24.00	29.00	841	35.04
53	34.40	18.60	345.96	10.06
45	25.60	19.40	376.36	14.7
2	40.00	38.00	1444.00	36.1

Chi Square = 125.516

http://math.hws.edu/javamath/ryan/ChiSquare.html

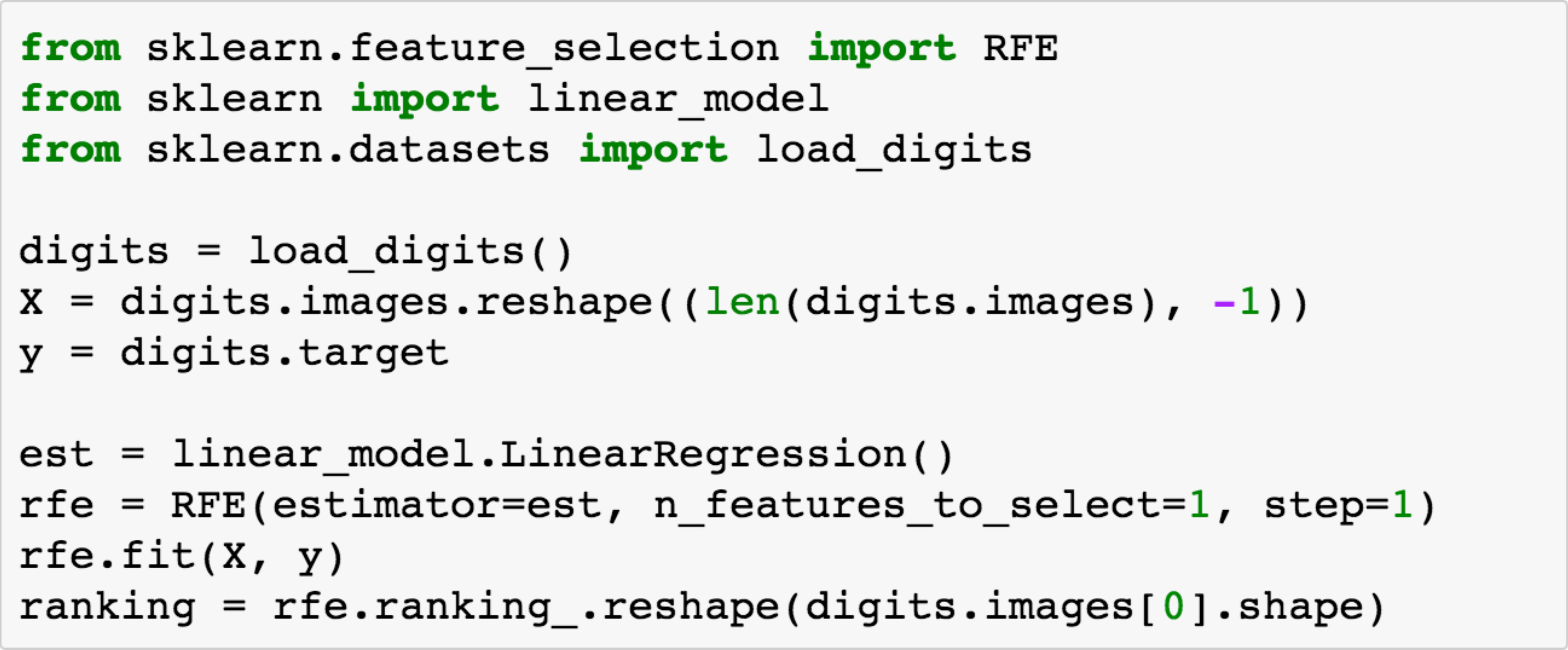
X^2 Test of Independence

- Similar to F-Test, Chi-Squares measure must be compared against a lookup table to determine p-value (statistical significance of the chi-square numbers).
- You can then use this p-value to assess importance of your features.
- Note that Chi-Squared works best when both features and targets are categorical. There is no continuous version. Features must be non-negatives, such as booleans or frequencies.
- more info: http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.chi2.html

Wrapper Methods

- **Recursive Feature Elimination** is an example of wrapper methods for feature selection.
- Given an external estimator that assigns weights to features (e.g., the coefficients of a linear model)
- Recursive feature elimination (RFE) is to select features by recursively considering smaller and smaller sets of features.
- First, the estimator is trained on the initial set of features and the importance of each feature is obtained either through a coef_ attribute or through a feature_importances_ attribute.
- Then, the least important features are pruned from current set of features.That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached.

Wrapper Methods



```
from sklearn.feature_selection import RFE
from sklearn import linear_model
from sklearn.datasets import load_digits

digits = load_digits()
X = digits.images.reshape((len(digits.images), -1))
y = digits.target

est = linear_model.LinearRegression()
rfe = RFE(estimator=est, n_features_to_select=1, step=1)
rfe.fit(X, y)
ranking = rfe.ranking_.reshape(digits.images[0].shape)
```

Feature Selection Trap

- It is important to consider feature selection a part of the model selection process. If you do not, it may lead to overfitting.
- Include feature selection within the inner-loop when you do cross-validation.
- A mistake would be to perform feature selection first to prepare your data, then perform model selection and training on the selected features.
- If you perform feature selection on all of the data and then cross-validate, then the test data in each fold of the cross-validation procedure was also used to choose the features and this is what biases the performance analysis.