Outlier handling:

So diff from main data

Suspectile:

Linear models

Ada boost

Handling: trimming

Missing data

Discretization

Censoring

Trimming: removing outliers from dataset

Missing data: treating outliers as missing data and perform imputations

Discretization: put outliers into lower and upper bins

Censoring: capping, top/bottom coding/ winsorization.

trimmingg: fast and remove big chunk of data

caping: no data removed , distorts variable distribution

how to detect:

guassian distribution

interquatile range proximity rule

quantile

Guassian:

Mean+/- 3\*sd

IQR:

Skewed:

Upperlimit: 75th quantile+1.5IQR

Lower limit= 25th quantile-1.5IQR

Normal distri: above and below 95th and 5th quantiles

**Outlier Engineering**

An outlier is a data point which is **significantly different from the remaining data**. “An outlier is an observation which **deviates so much from the other observations as** to arouse suspicions that it was generated by a different mechanism.” [D. Hawkins. Identification of Outliers, Chapman and Hall , 1980].

Statistics such as the mean and variance are very susceptible to outliers. In addition, **some Machine Learning models are sensitive to outliers** which may decrease their performance. Thus, depending on which algorithm we wish to train, we often remove outliers from our variables.

We discussed in section 3 of this course how to identify outliers. In this section, we we discuss how we can process them to train our machine learning models.

**How can we pre-process outliers?**

* Trimming: remove the outliers from our dataset
* Treat outliers as missing data, and proceed with any missing data imputation technique
* Discrestisation: outliers are placed in **border bins together** with higher or lower values of the distribution
* Censoring: **capping the variable distribution** at a max and / or minimum value

**Censoring** is also known as:

* top and bottom coding
* winsorization
* capping

**Trimming or truncation**

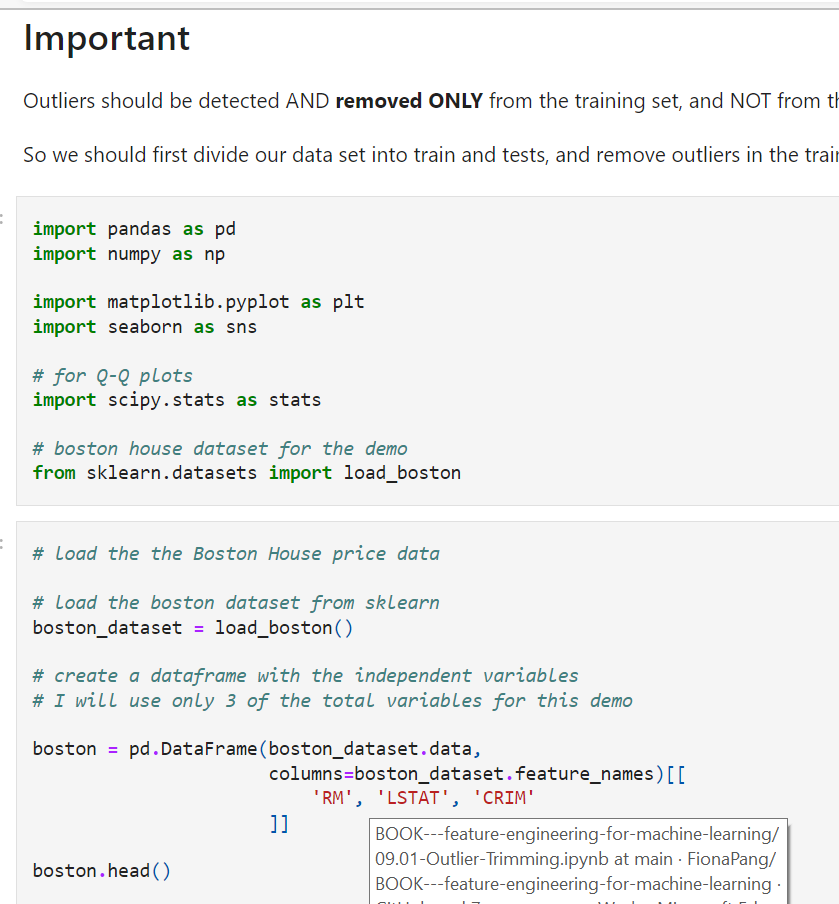
Trimming, also known **as truncation**, **involves removing the outliers from the dataset**. We only need to decide on a metric to determine outliers. As we saw in section 3, this can be the Gaussian approximation for normally distributed variables or the inter-quantile range proximity rule for skewed variables.

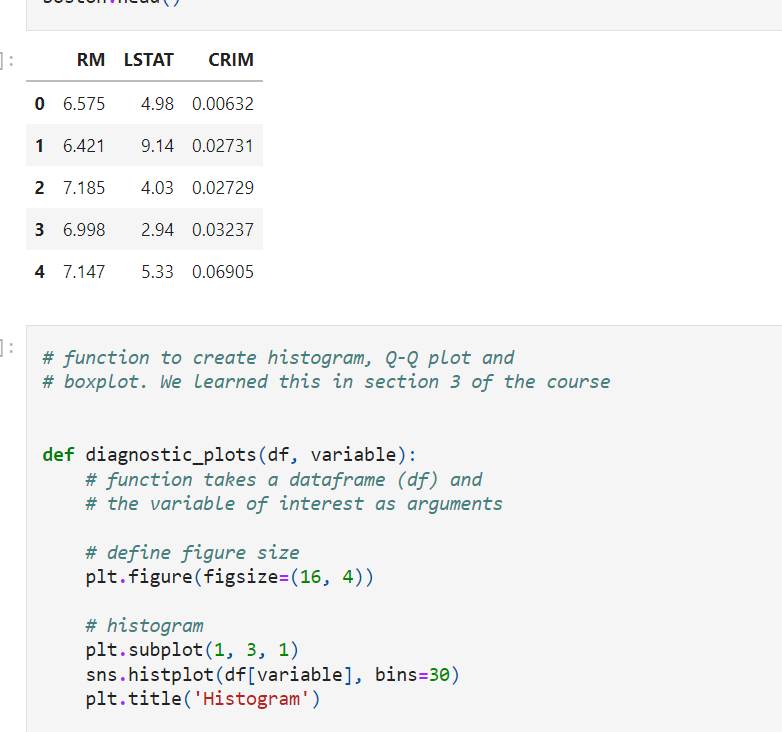
**Advantages**

* quick

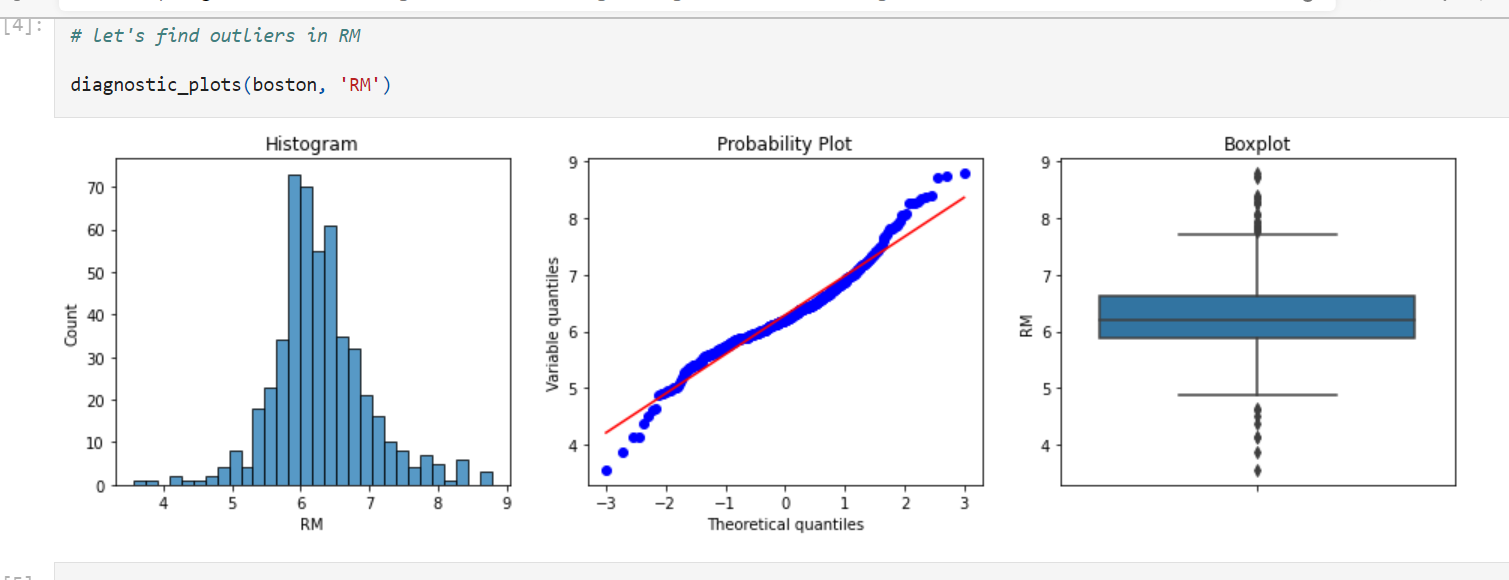
**Limitations**

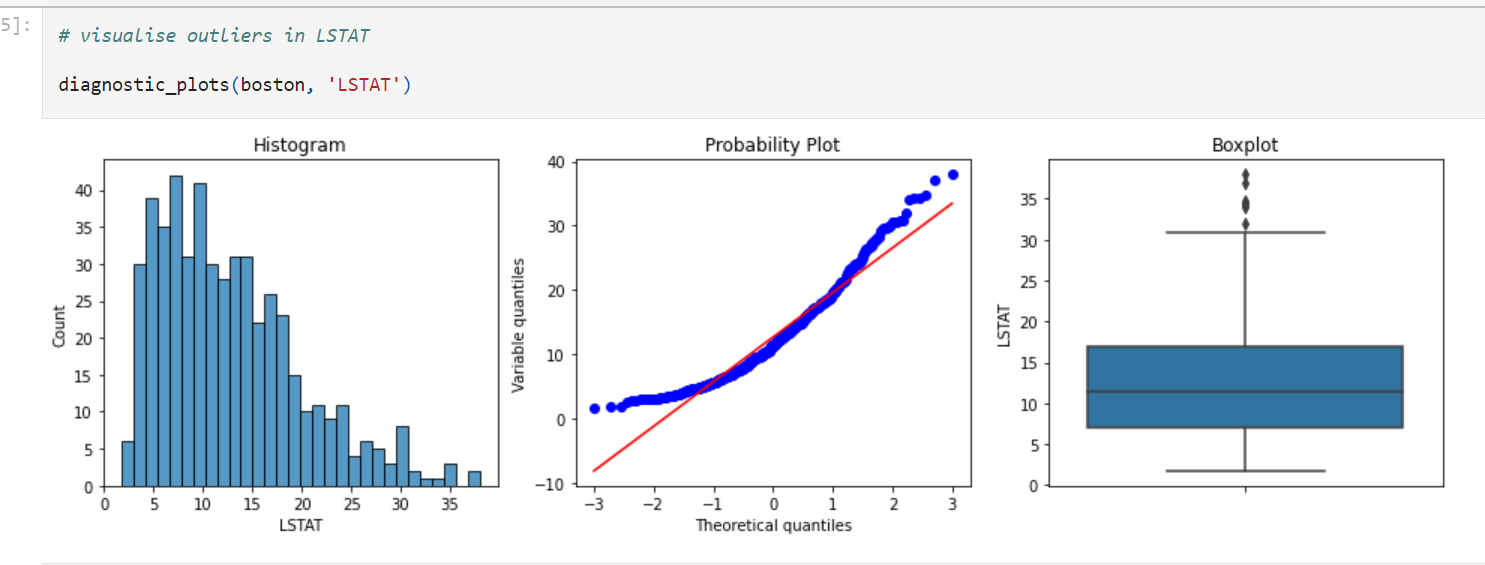
* **outliers for one variable could contain useful information** in the other variables
* if there are outliers across many variables, we could remove a **big chunk of dataset**

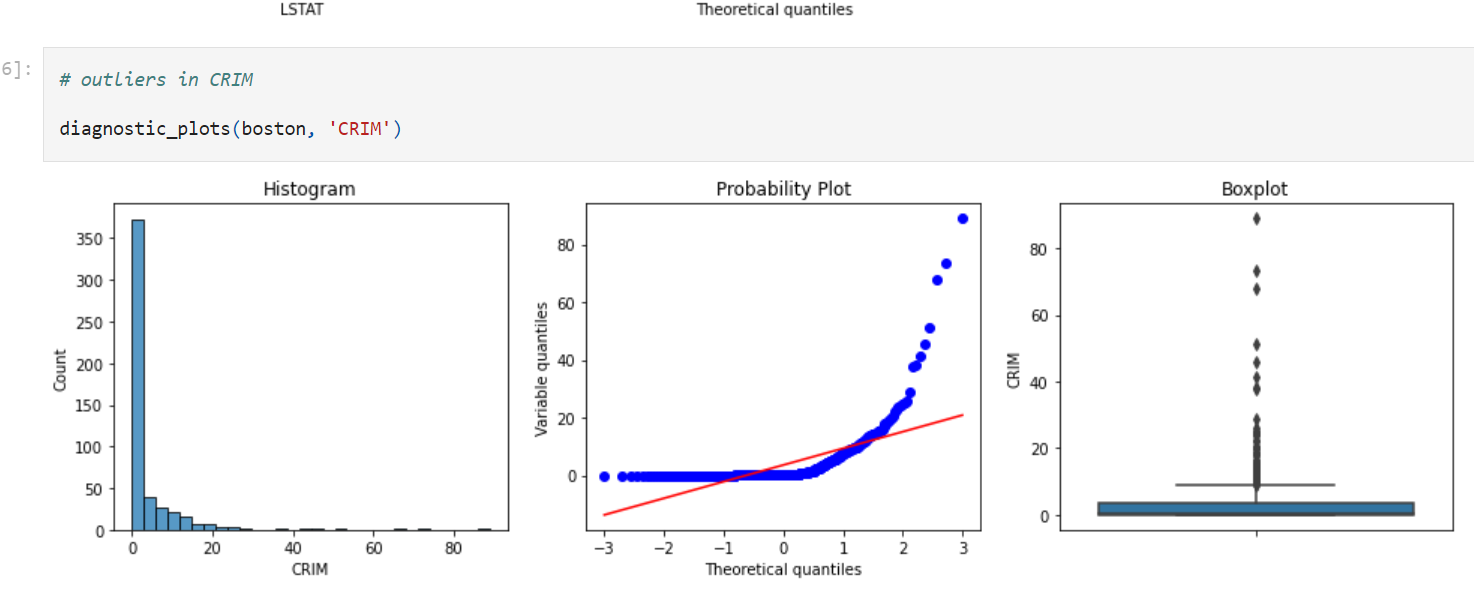






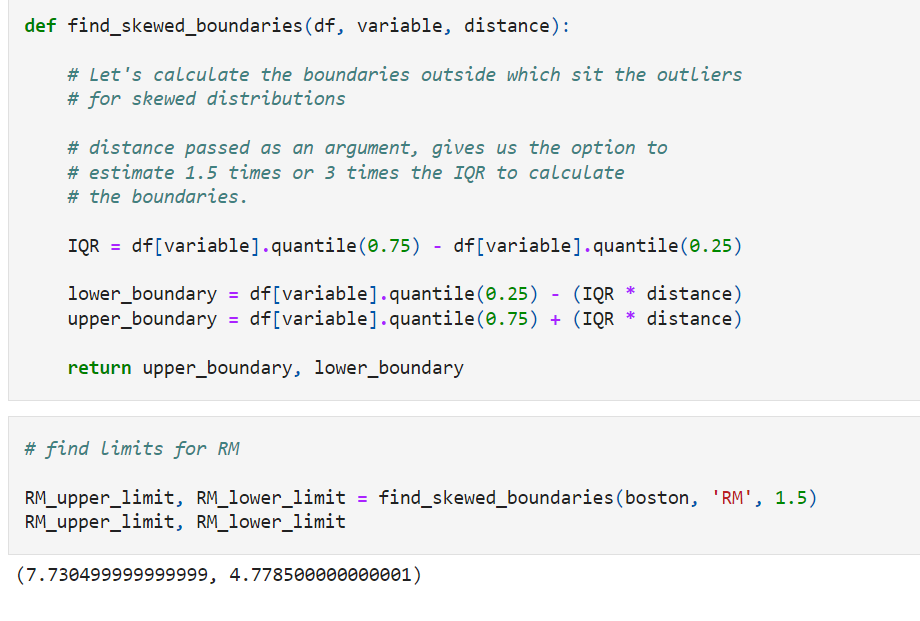




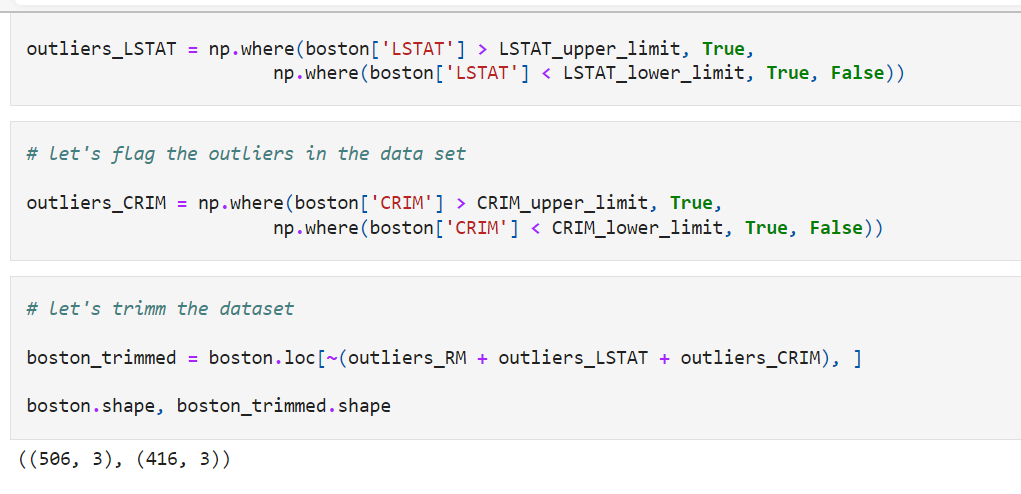


There are outliers in all of the above variables. RM shows outliers in both tails, whereas LSTAT and CRIM only on the right tail.

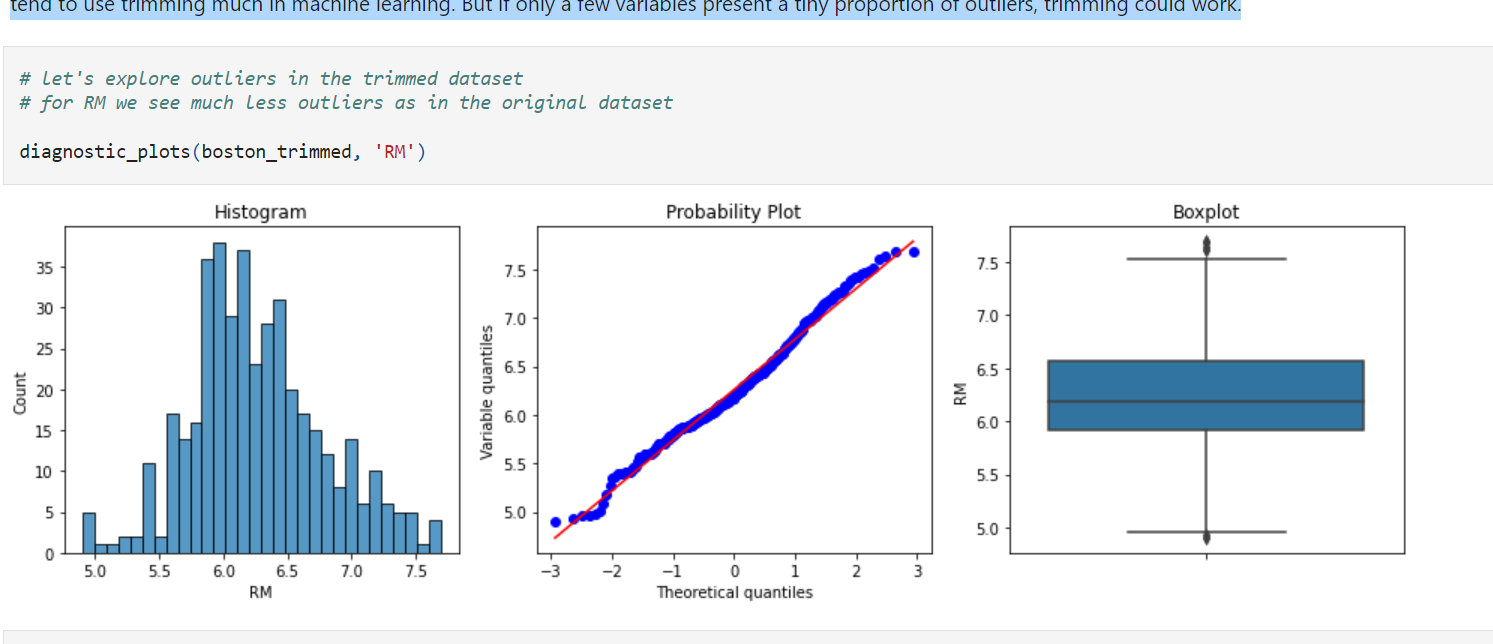
To find the outliers, let's re-utilise the function we learned in section 3:

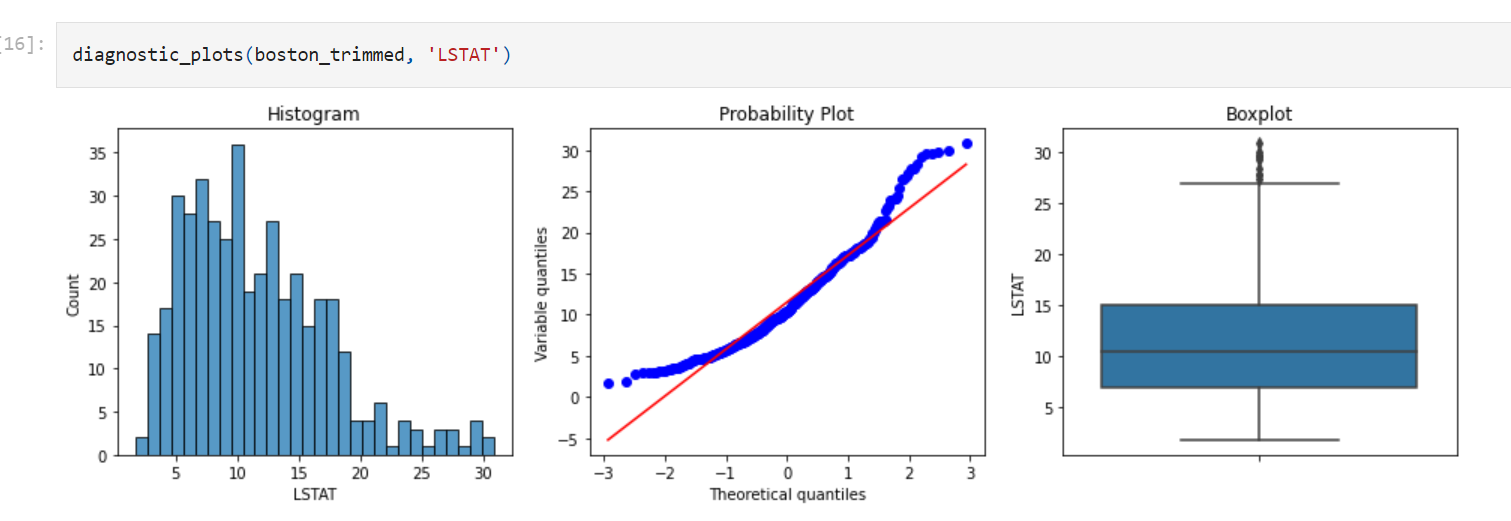


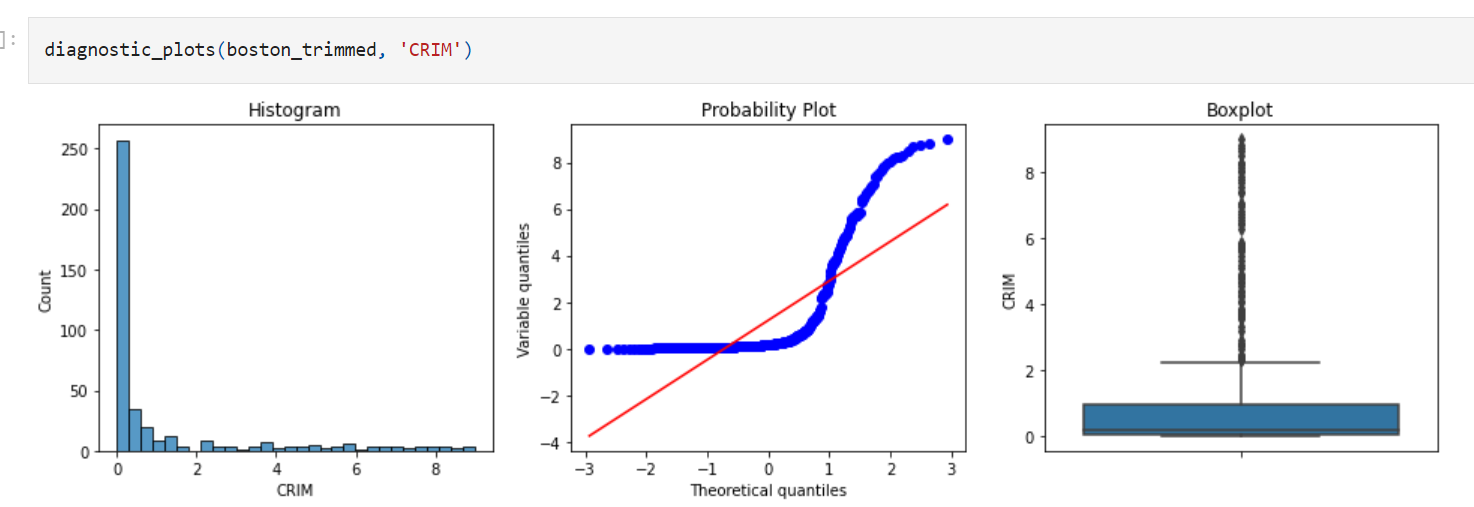




We can see that using trimming, we removed almost 100 rows, from a dataset of 500 rows, this is about 20% of the data was removed. This is mostly why, we do not tend to use trimming much in machine learning. But if only a few variables present a tiny proportion of outliers, trimming could work.







For LSTAT and CRIM, we still see many outliers. When we remove data points from our dataset, all the parameters of the distribution are re-calculated, those are the mean, quantiles and inter-quantile range, therefore, in the new -trimmed- variable, values that before were not considered outliers, now are.

This is an unwanted characteristic of this way of coping with outliers.

**New: outlier trimming with Feature-engine**

Find out how to trim outliers with Feature-engine in the documentation:

<https://feature-engine.readthedocs.io/en/latest/outliers/OutlierTrimmer.html>

**Censoring or Capping.**

**Censoring**, or **capping**, **means capping the maximum and /or minimum of a distribution at an arbitrary value**.

On other words, values **bigger or smaller than the arbitrarily determined ones** are **censored**.

Capping can be done **at both tails,** or just one of the tails, depending on the variable and the user.

Check my talk in [pydata](https://www.youtube.com/watch?v=KHGGlozsRtA) for an example of capping used in a finance company.

**The numbers at which to cap the distribution can be determined**:

* arbitrarily
* using the inter-quantal range proximity rule
* using the gaussian approximation
* using quantiles

**Advantages**

* does not remove data

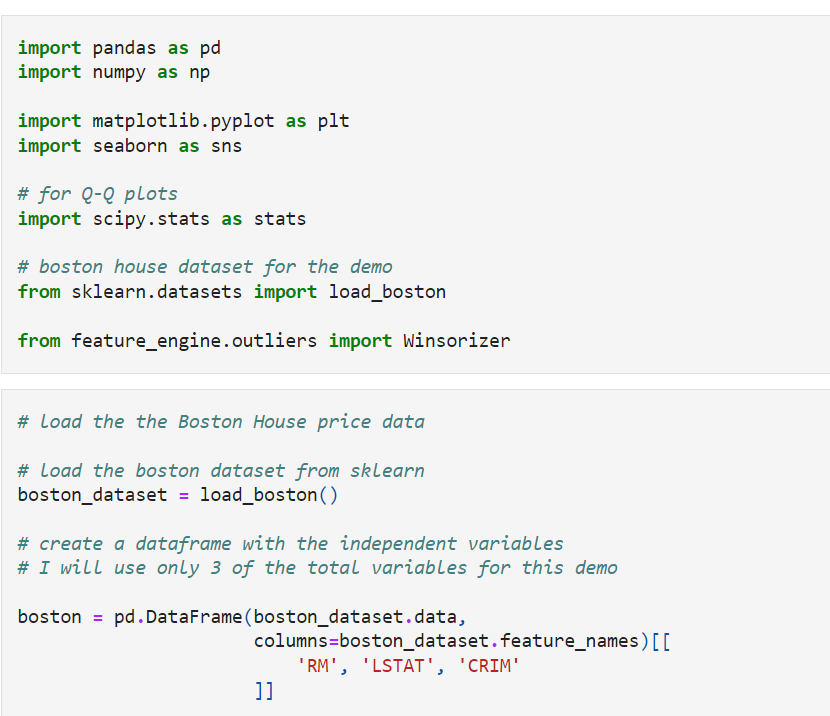
**Limitations**

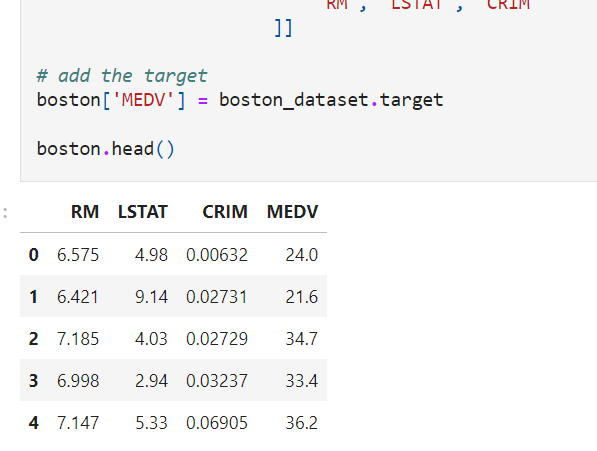
* distorts the distributions of the variables
* distorts the relationships among variables

**Important**

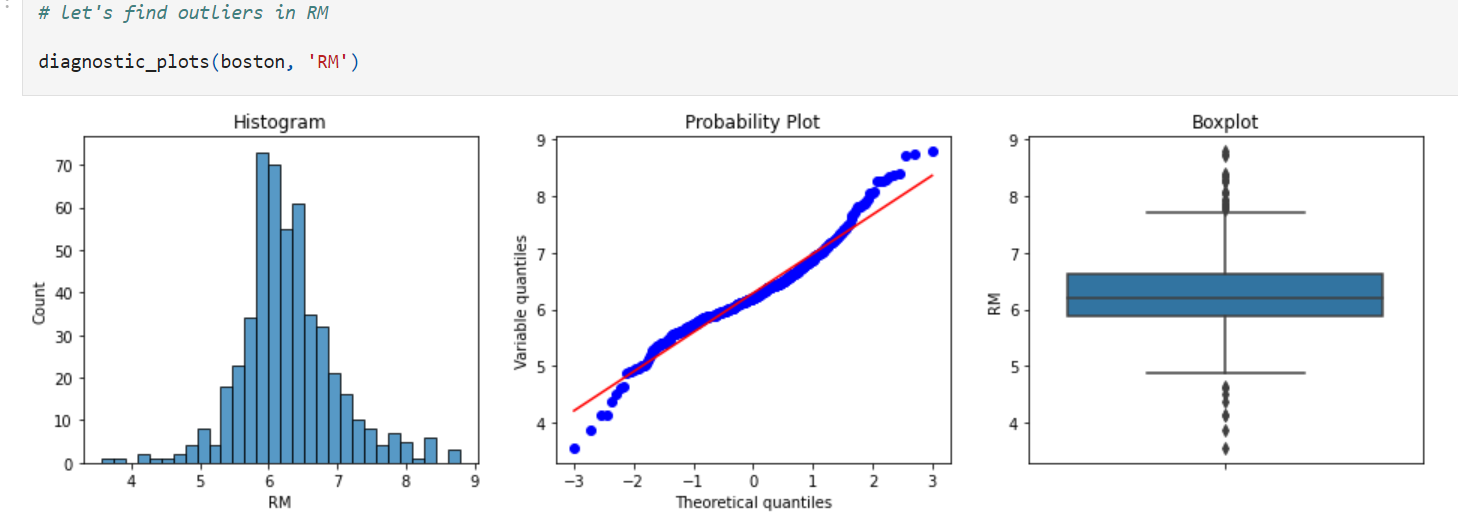
When **doing cappin**g, we tend to **cap values both in train and test set. It** is important to remember that the capping values **MUST be derived from the train set.** And then use those same values to cap the variables in the test set

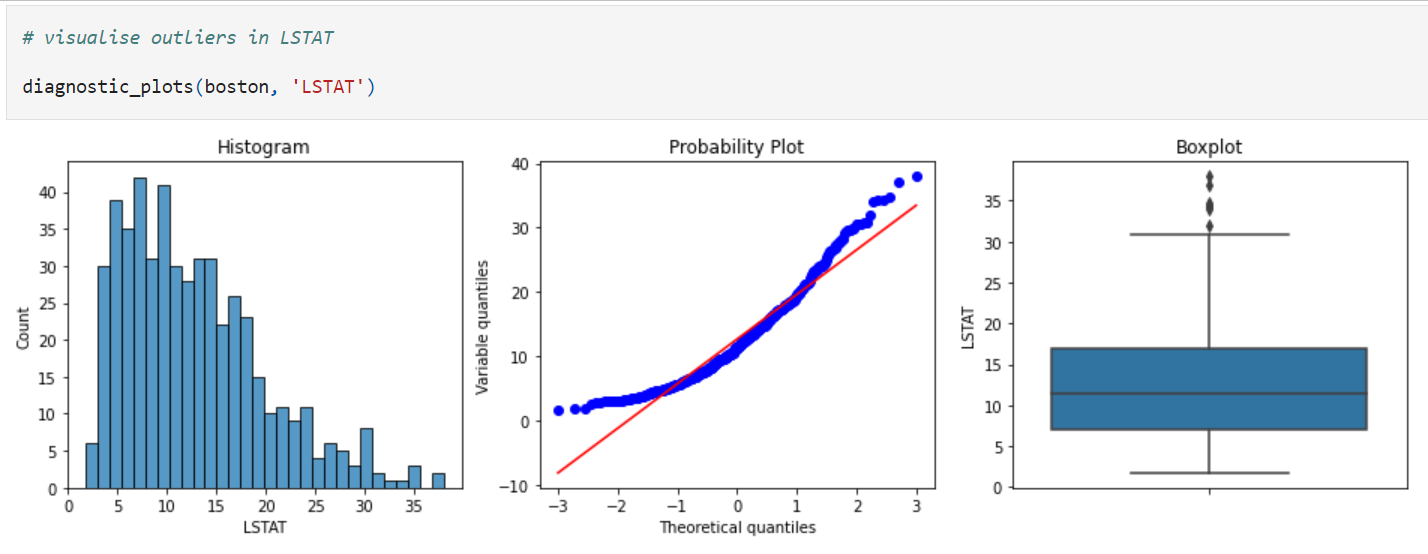
I will not do that in this demo, but please keep that in mind when setting up your pipelines

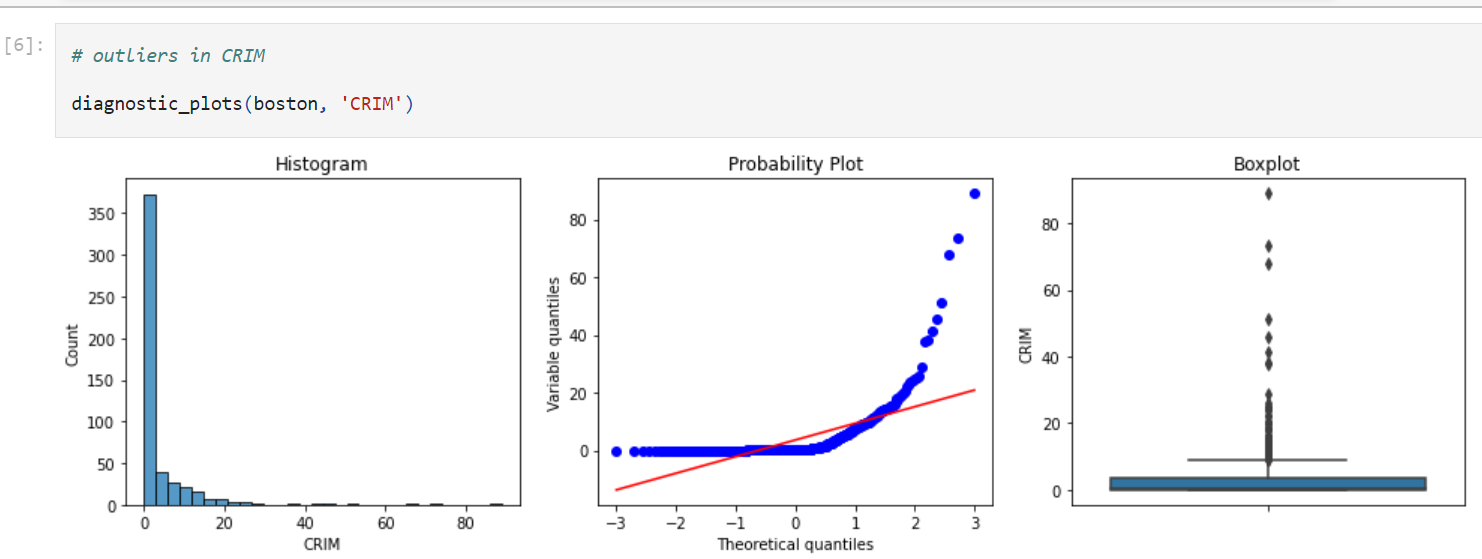






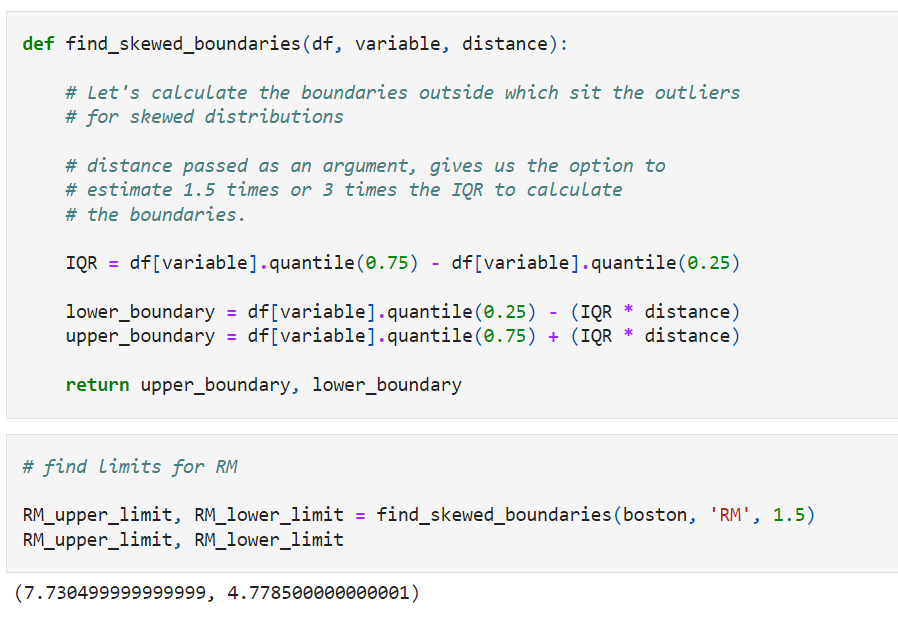


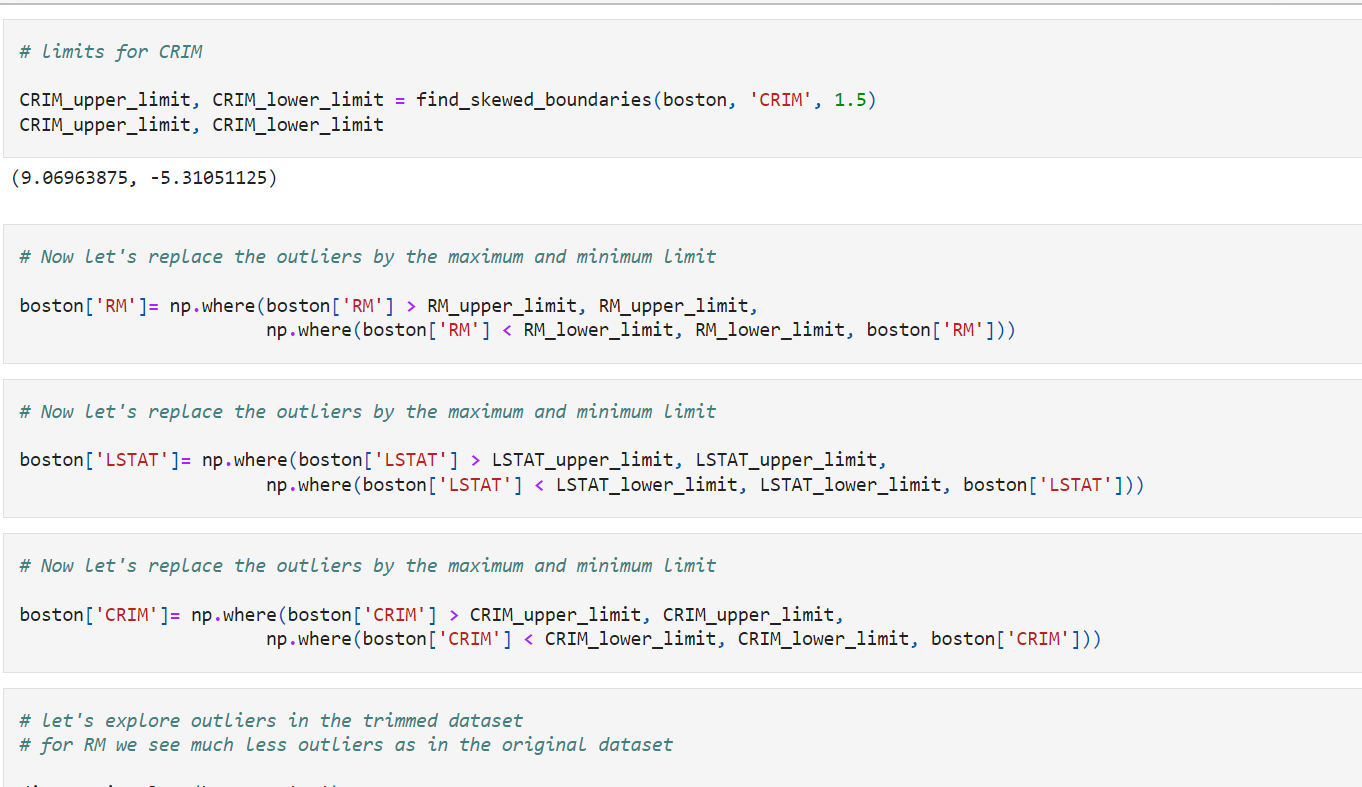


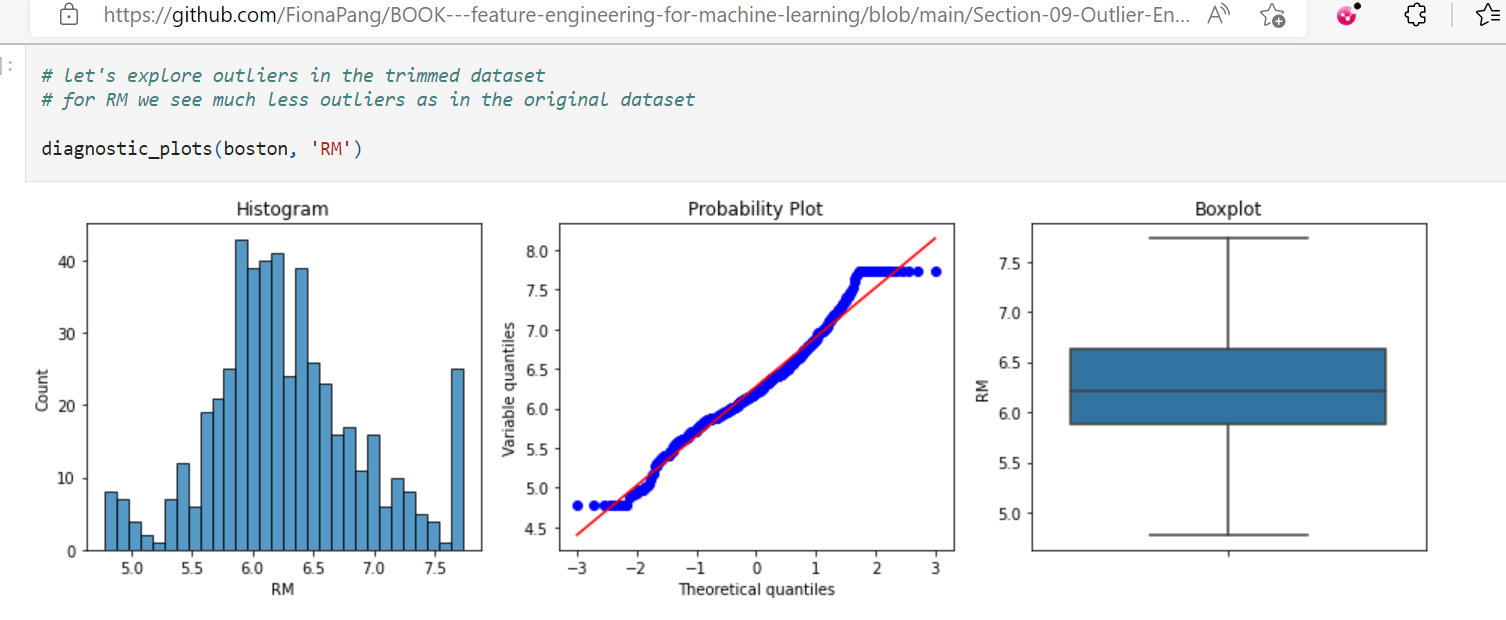


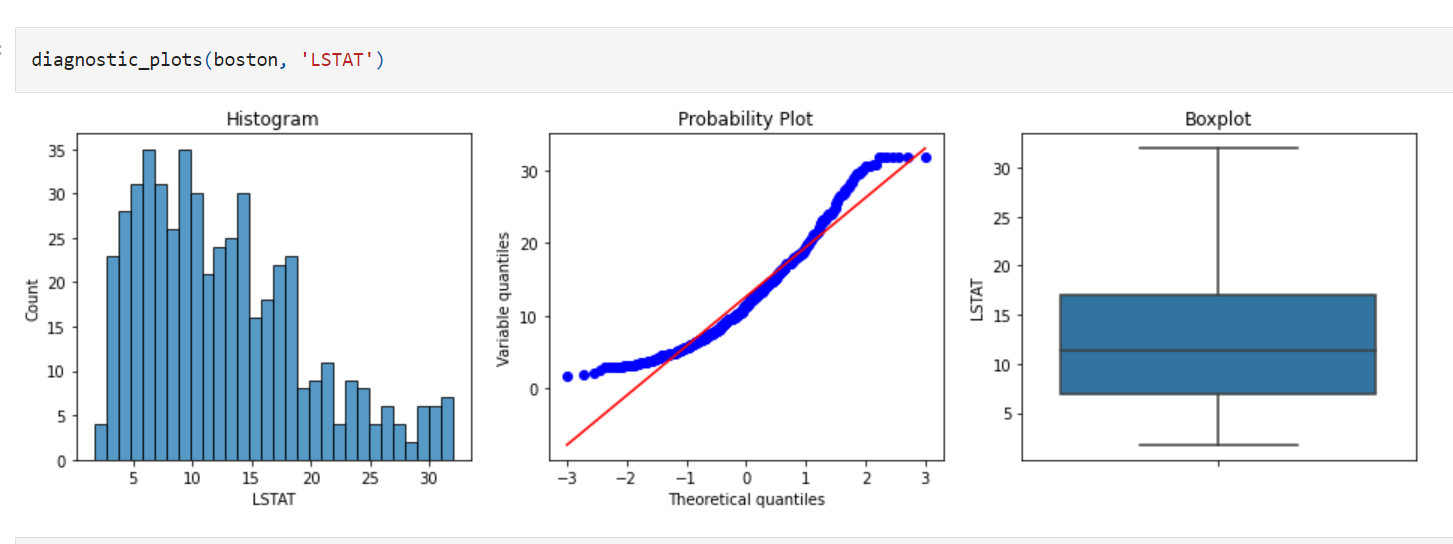
There are outliers in all of the above variables. RM shows outliers in both tails, whereas LSTAT and CRIM only on the right tail.

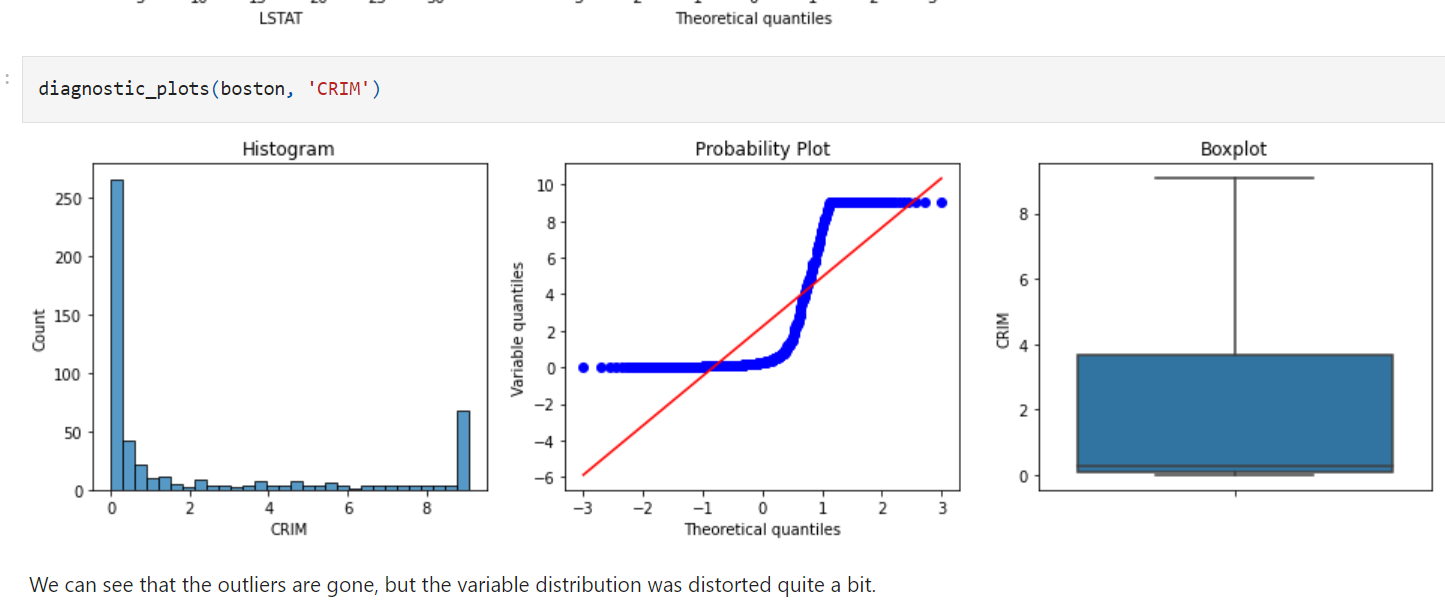
To find the outliers, let's re-utilise the function we learned in section 3:

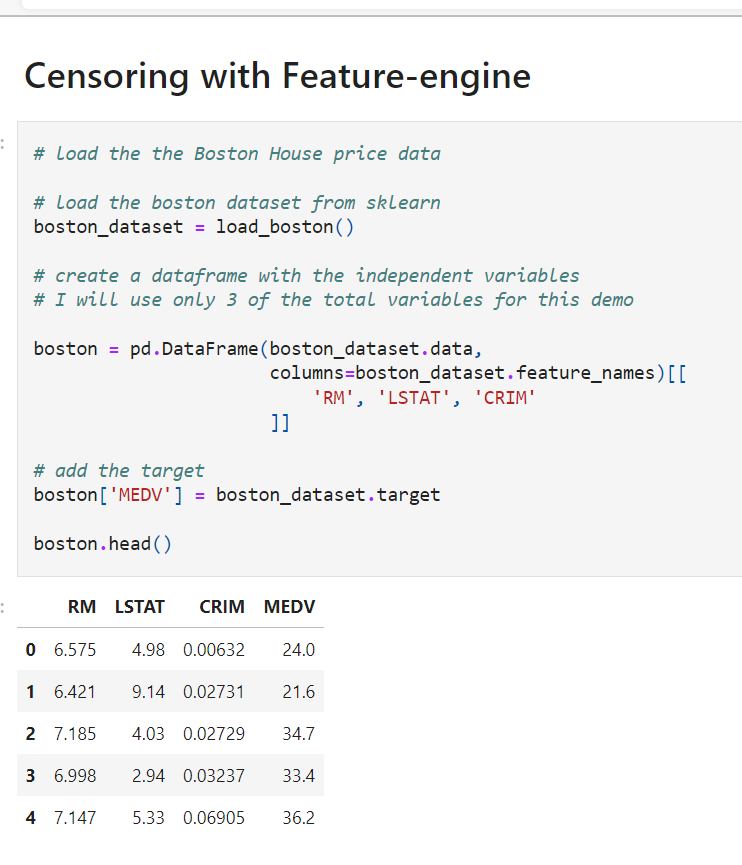




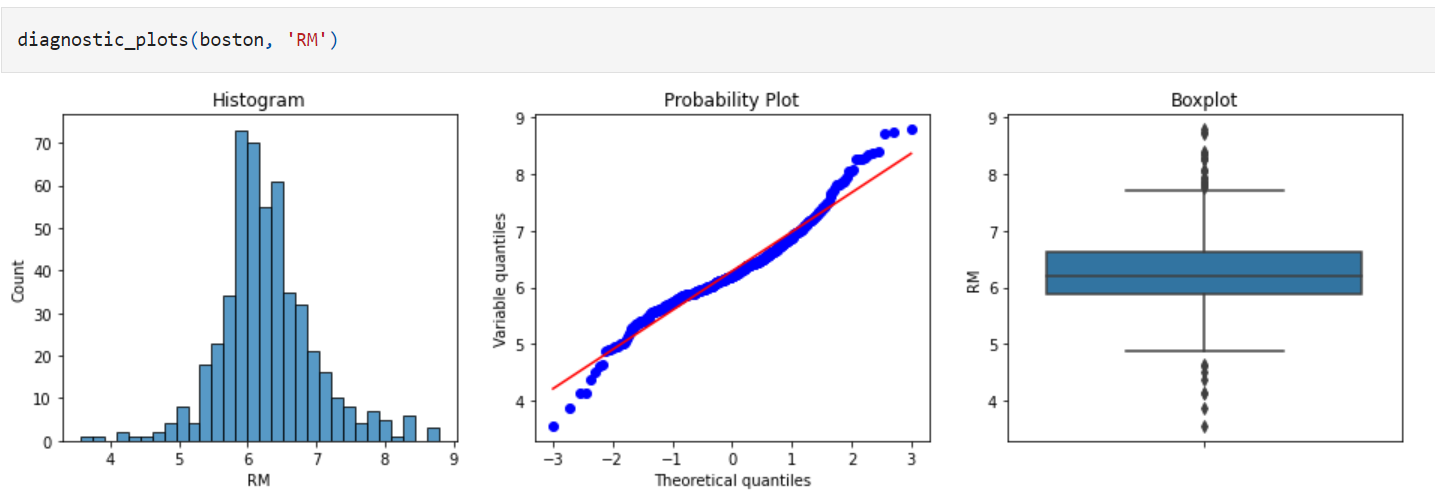


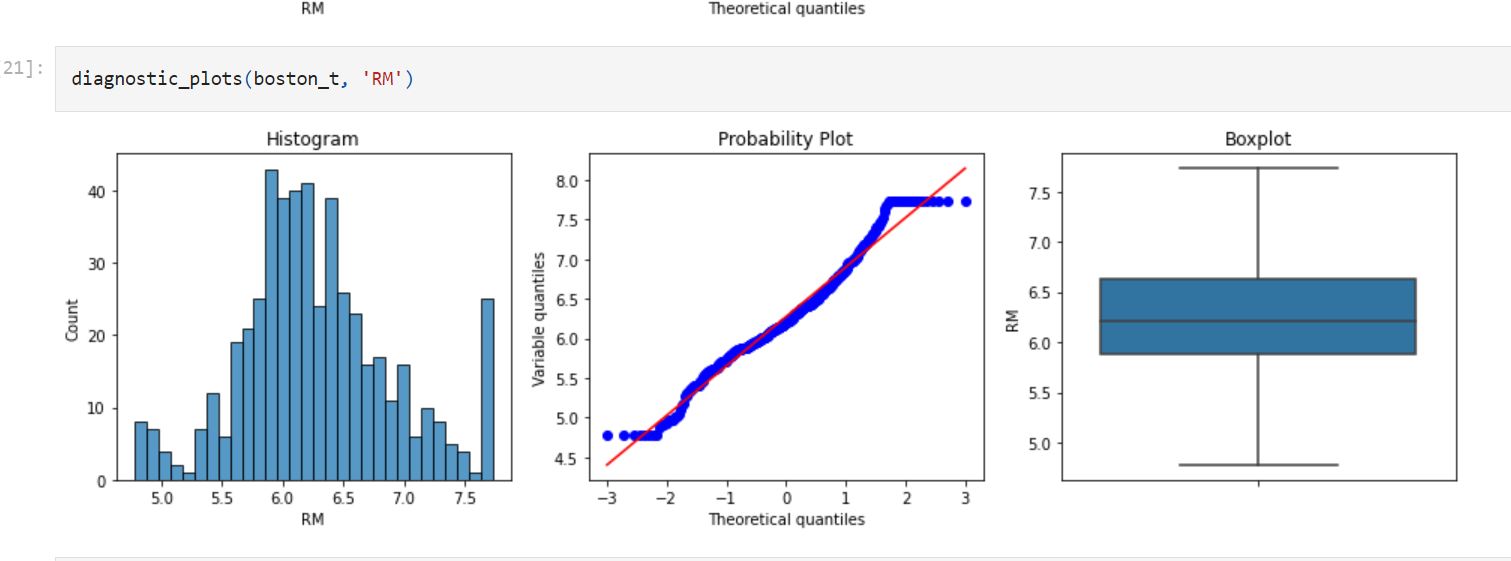


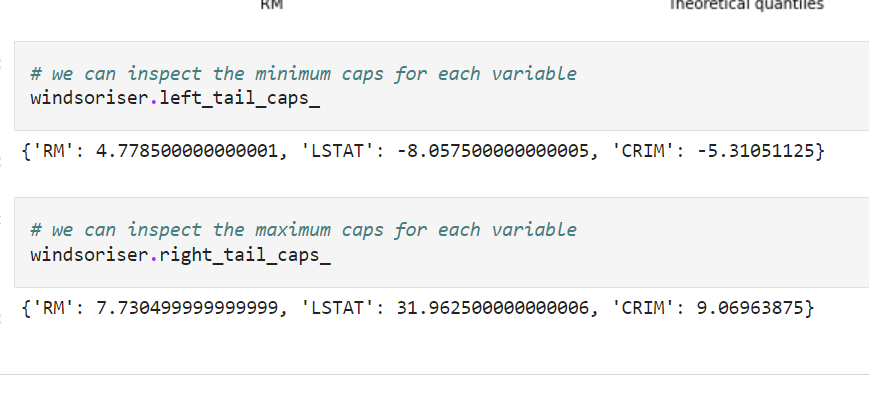












**Censoring or Capping.**

**Censoring**, or **capping**, means capping the maximum and /or minimum of a distribution at an arbitrary value. On other words, values bigger or smaller than the arbitrarily determined ones are **censored**.

Capping can be done at both tails, or just one of the tails, depending on the variable and the user.

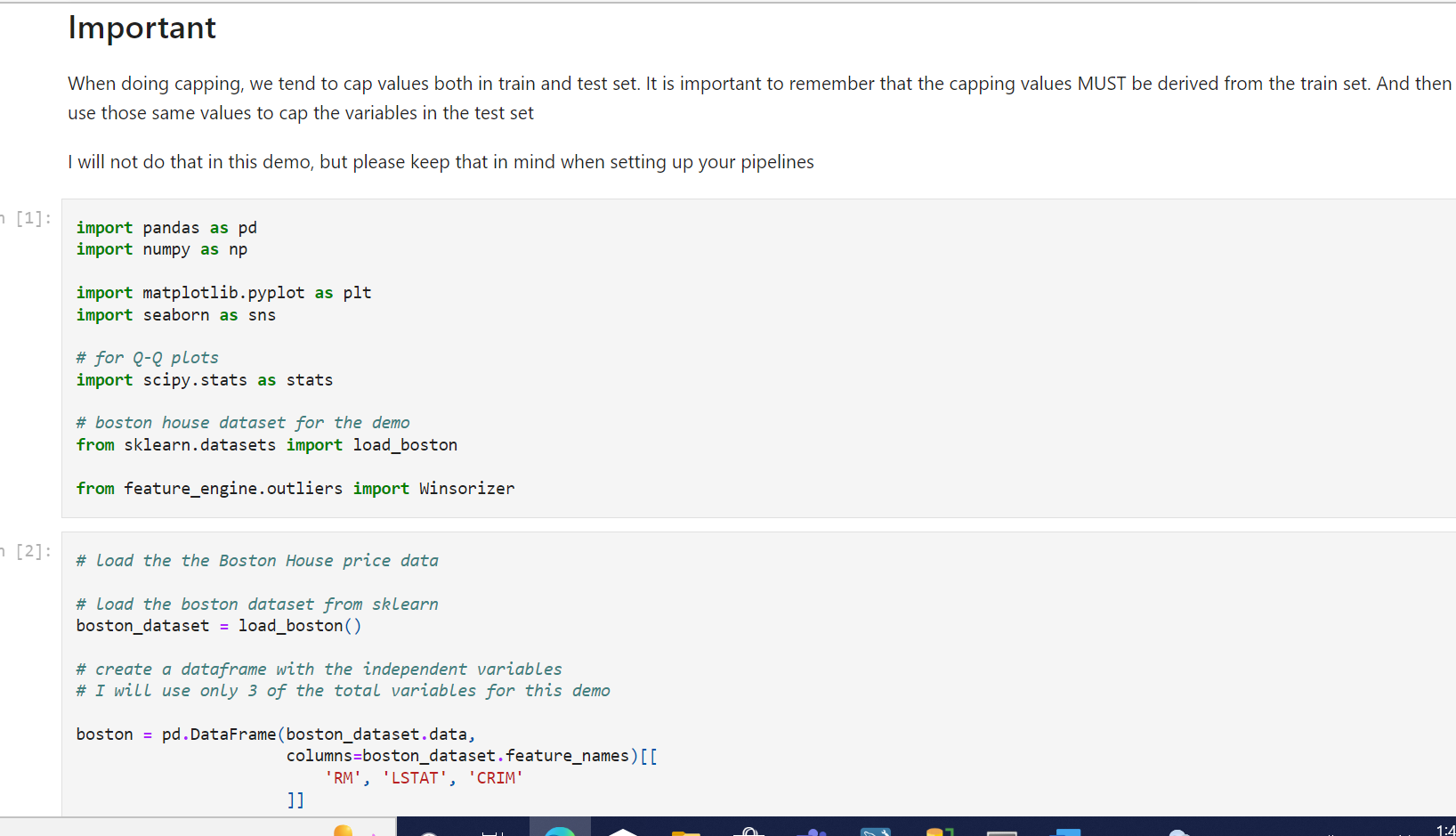
Check my talk in [pydata](https://www.youtube.com/watch?v=KHGGlozsRtA) for an example of capping used in a finance company.

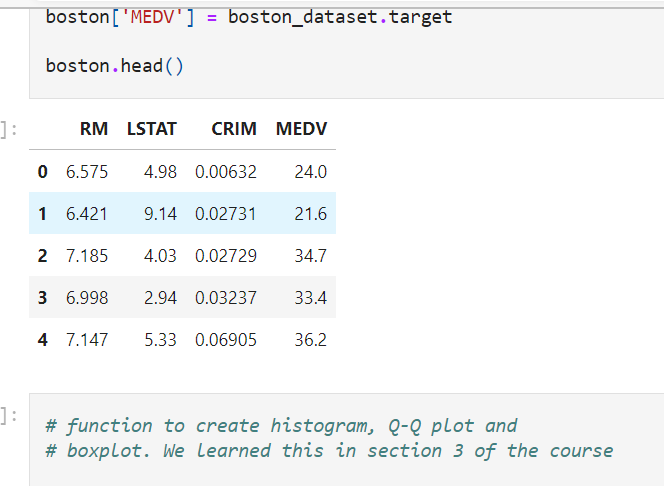
The numbers at which to cap the distribution can be determined:

* arbitrarily
* using the inter-quantal range proximity rule
* using the gaussian approximation
* using quantiles

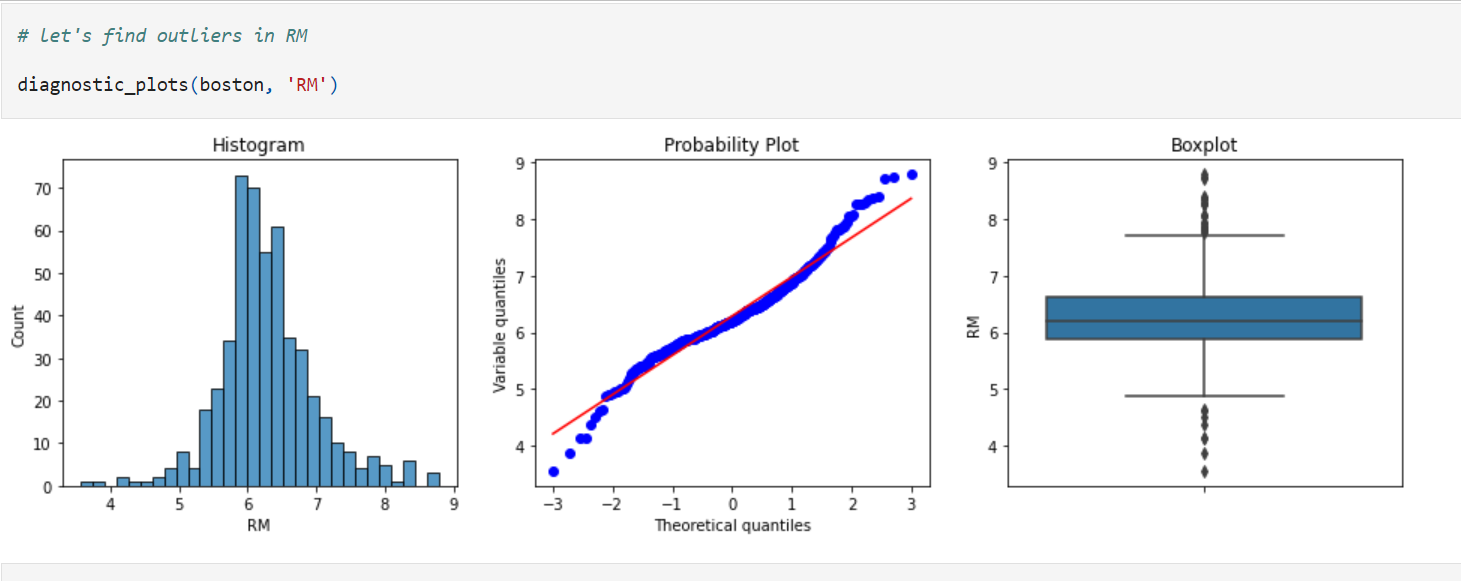
**Advantages**

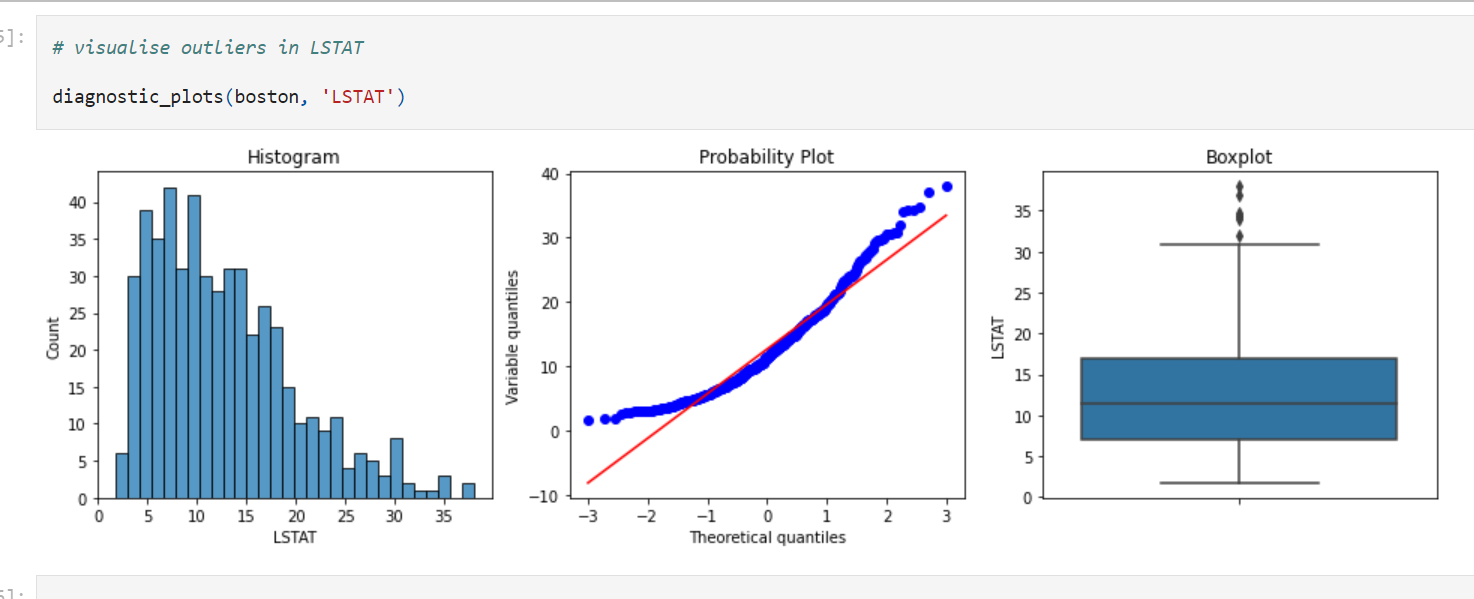
* does not remove data

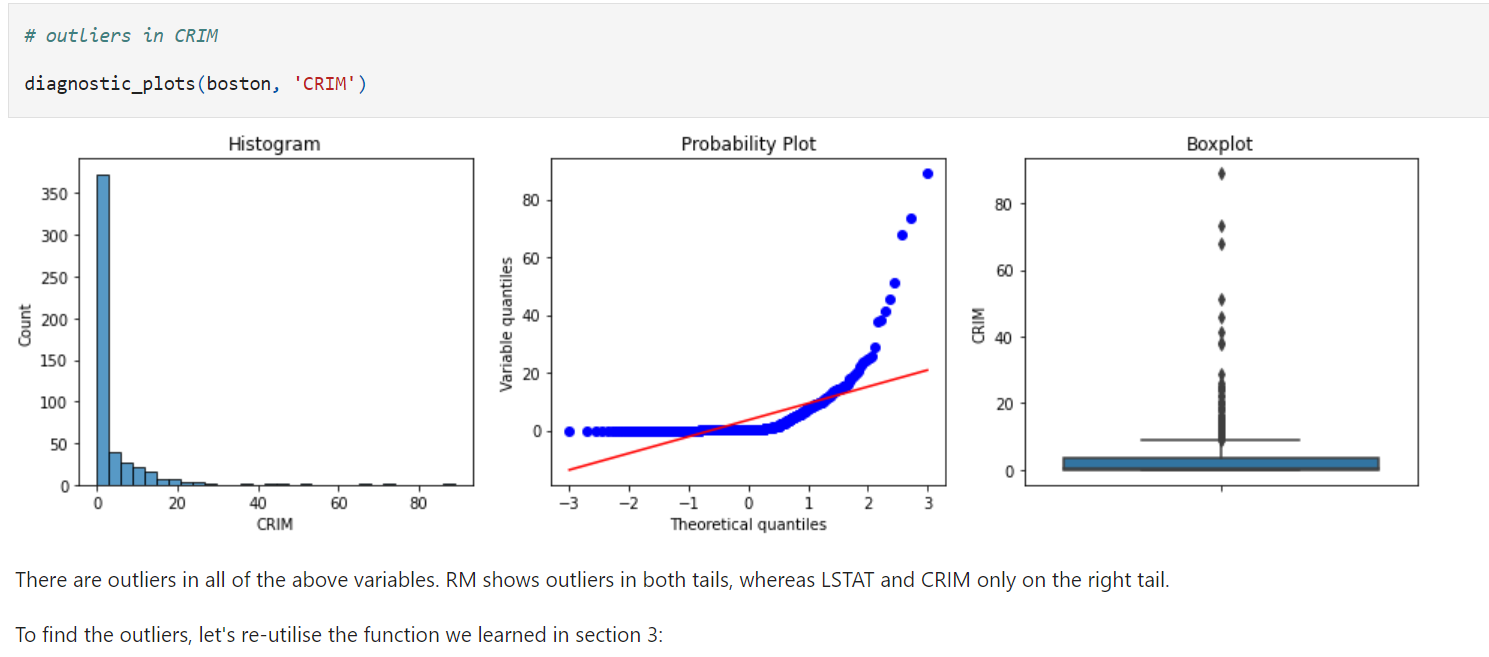


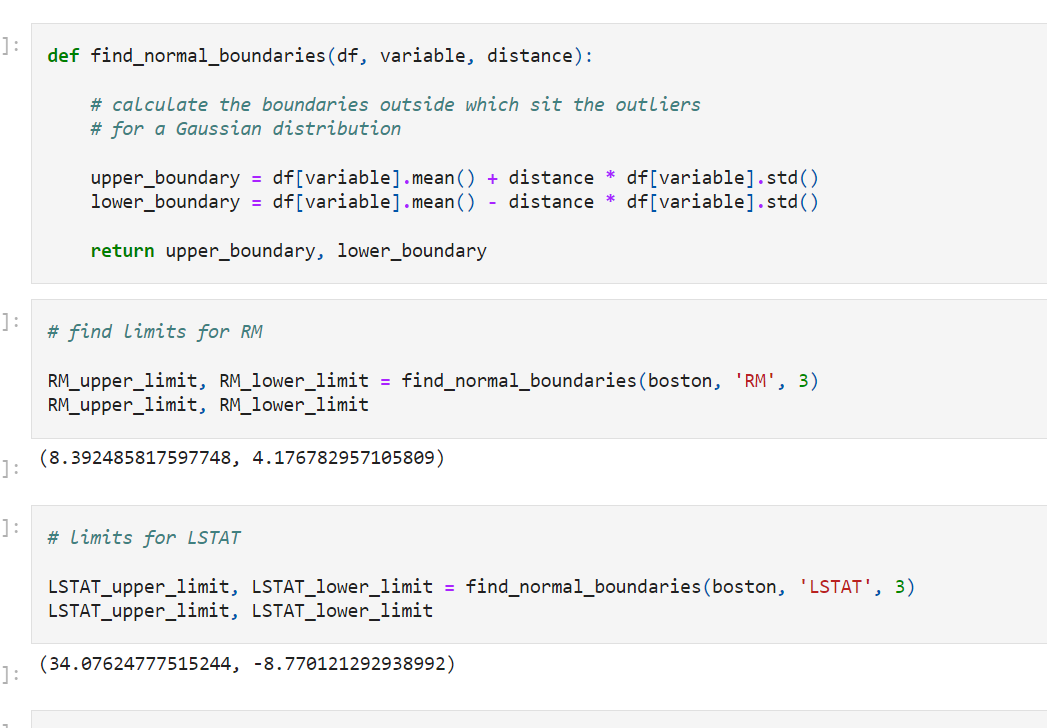


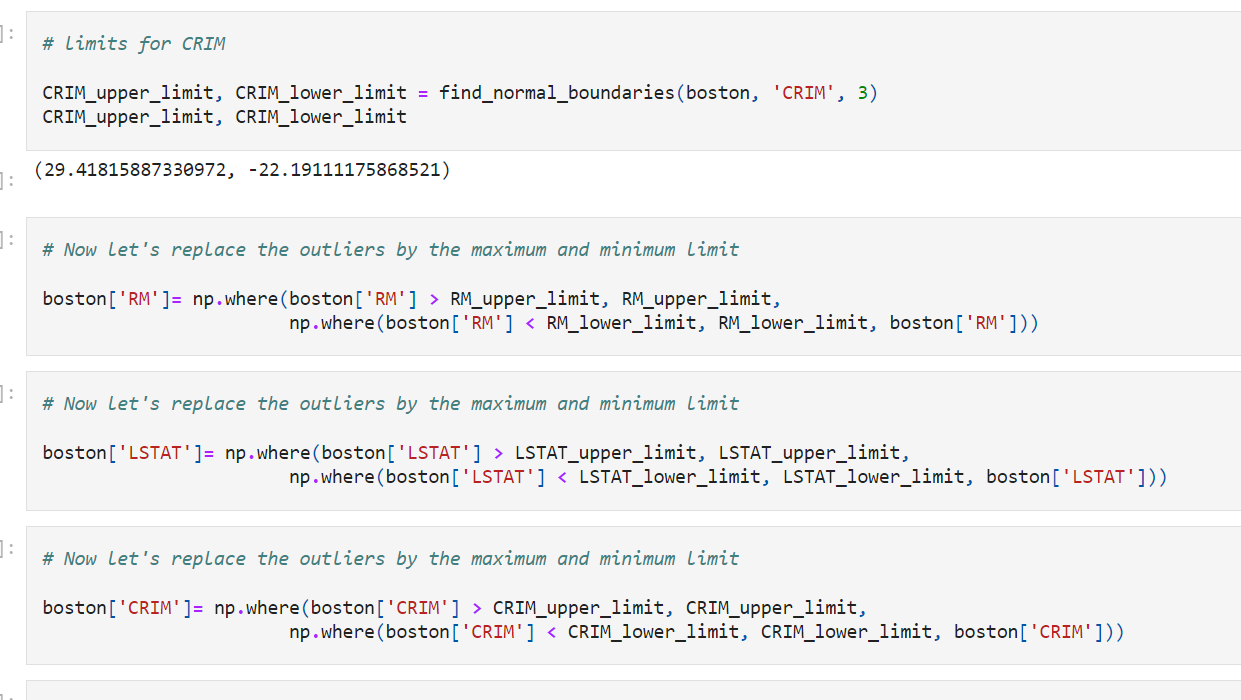


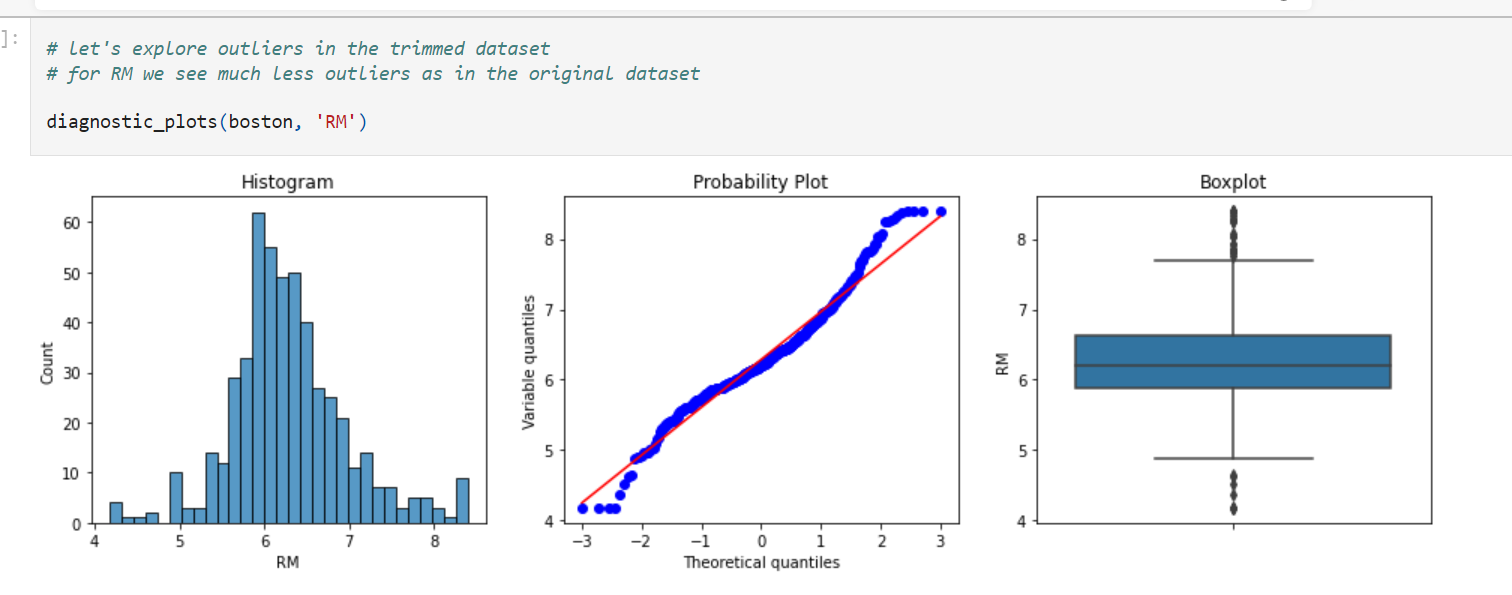


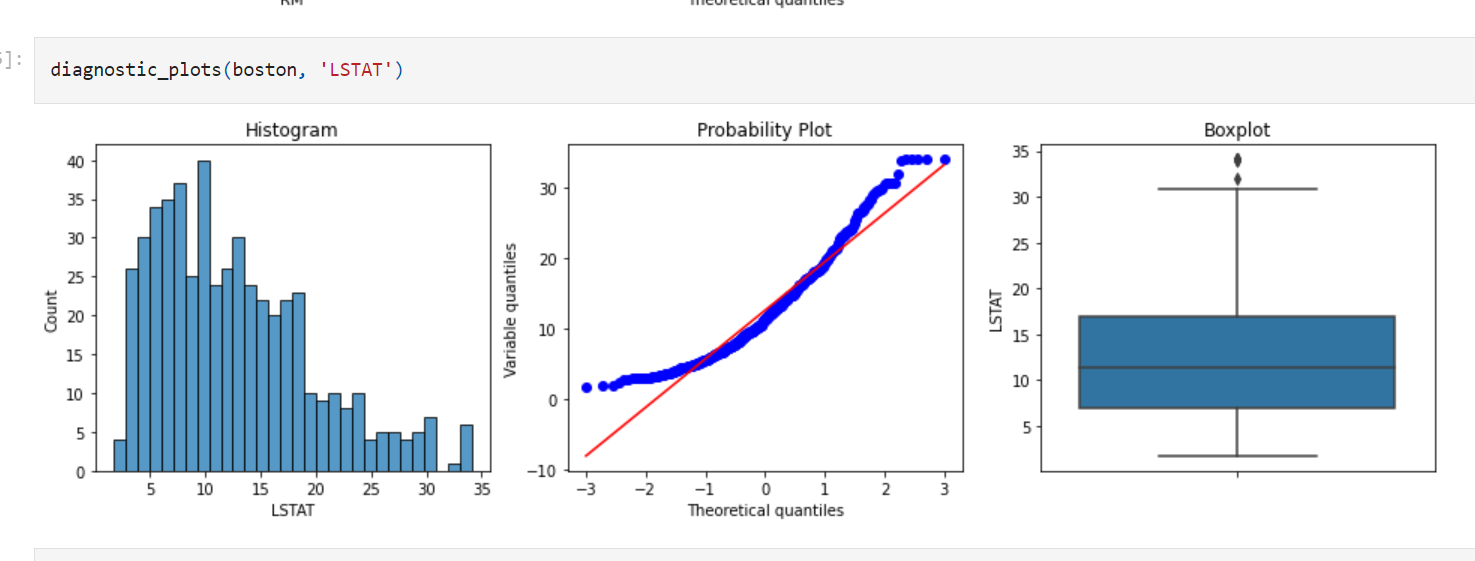


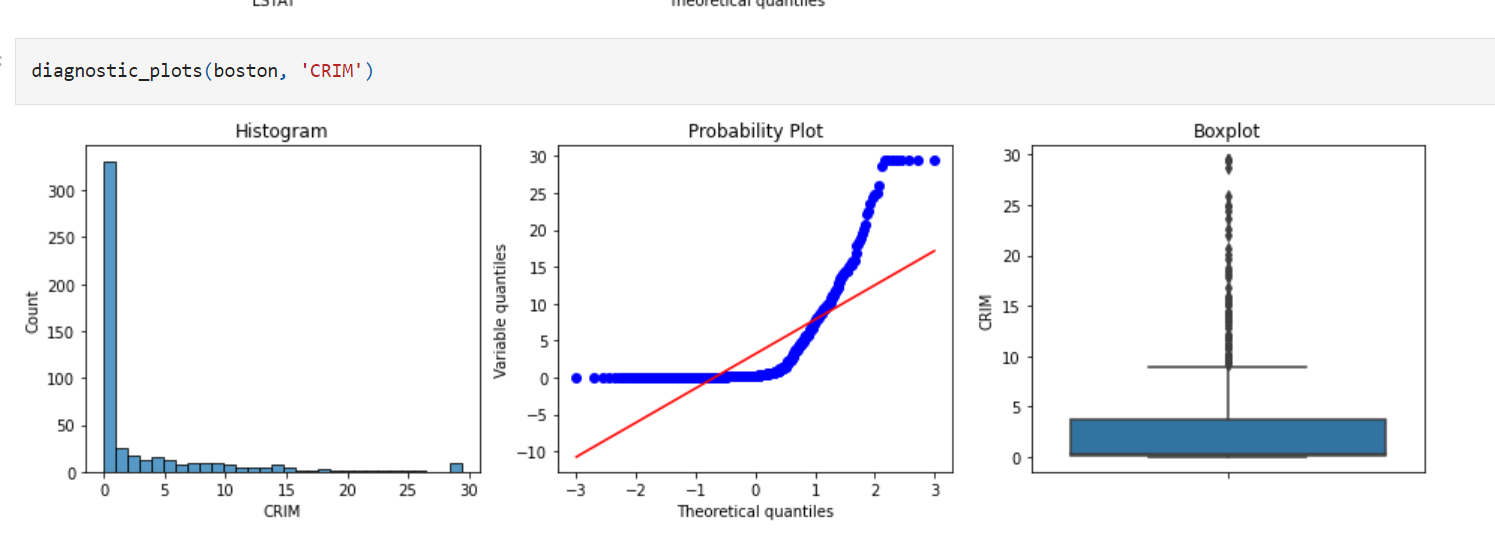






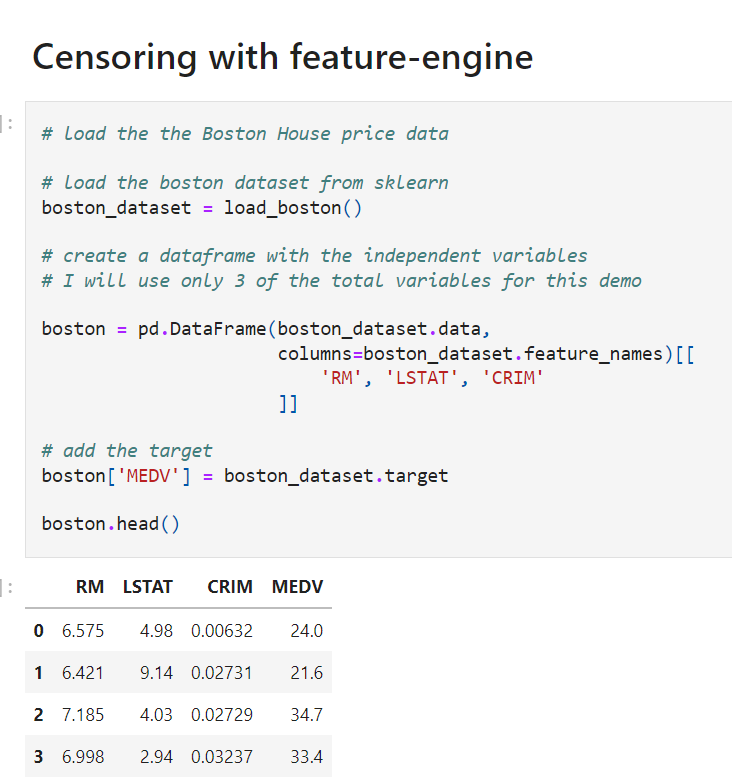


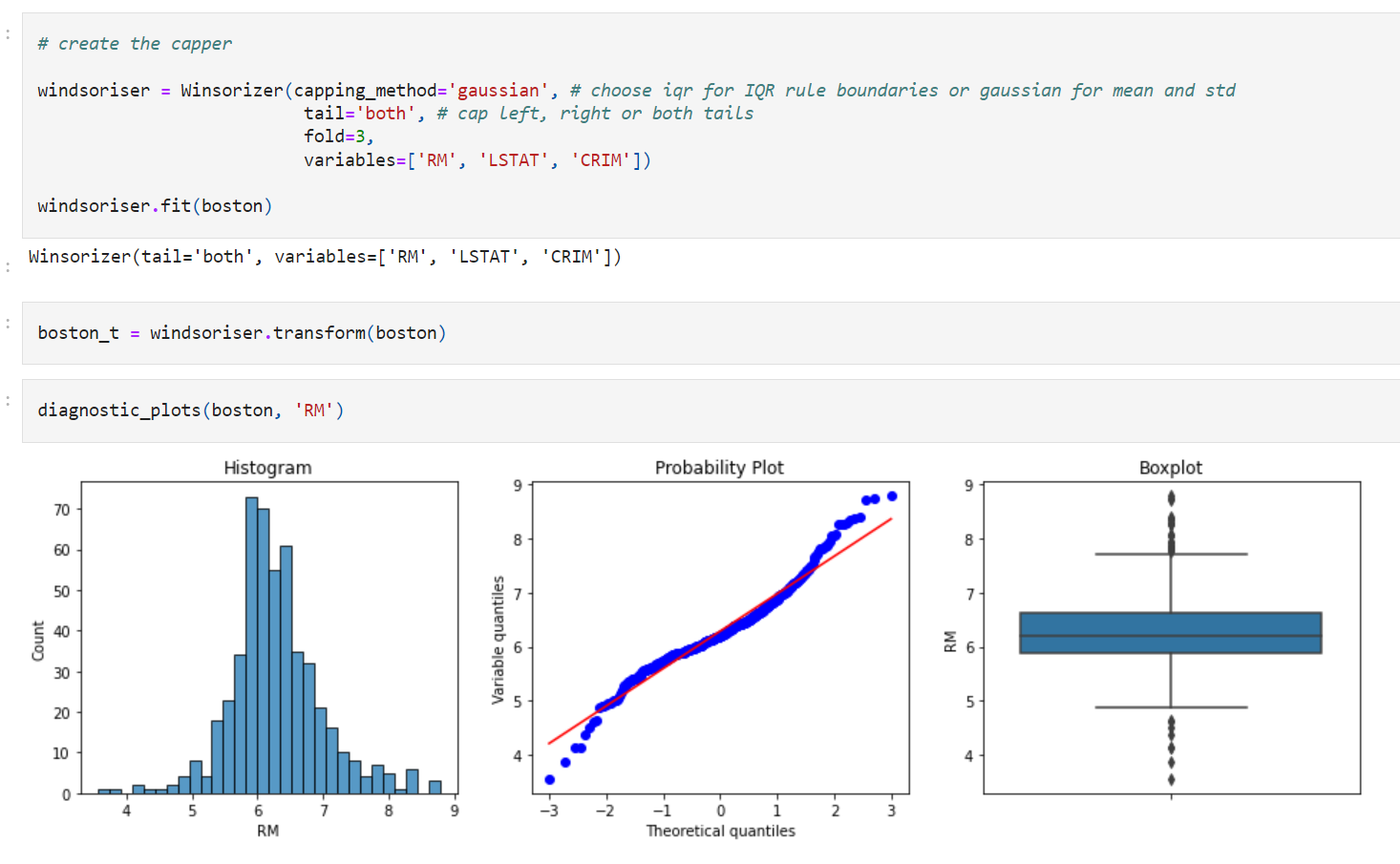


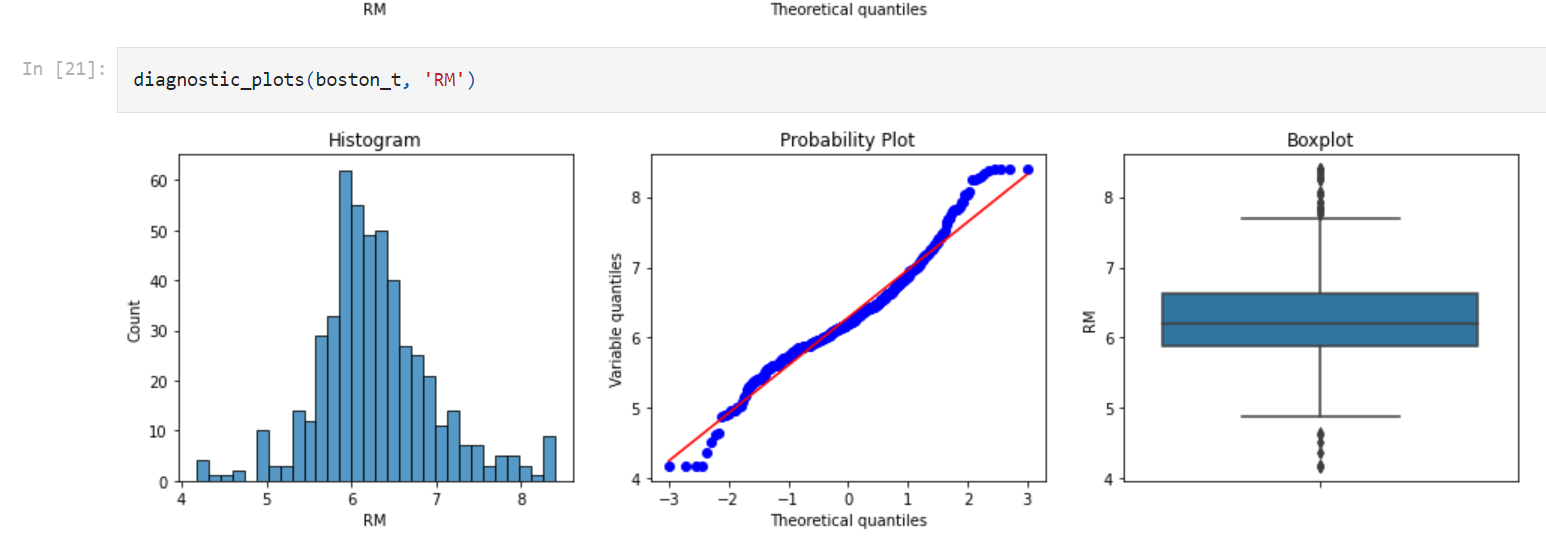


None of these variables are normally distributed, that is probably why the Gaussian approximation was not so effective to remove outliers. We could try and use a smaller distance, instead of multiplying by 3 times the std, we could 2 times or 1.5. But those numbers are set arbitrarily, and do not pose much statistical sense, therefore defeating the point of using the Gaussian approximation.

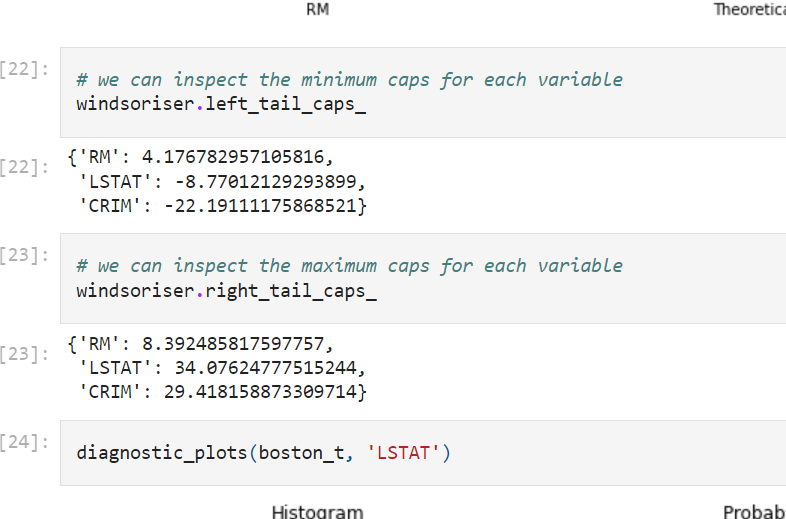
If this capping does not work as desired, I recommend you use the IQR rule as we discussed in the previous notebook, or quantiles, as we will see in the next one.

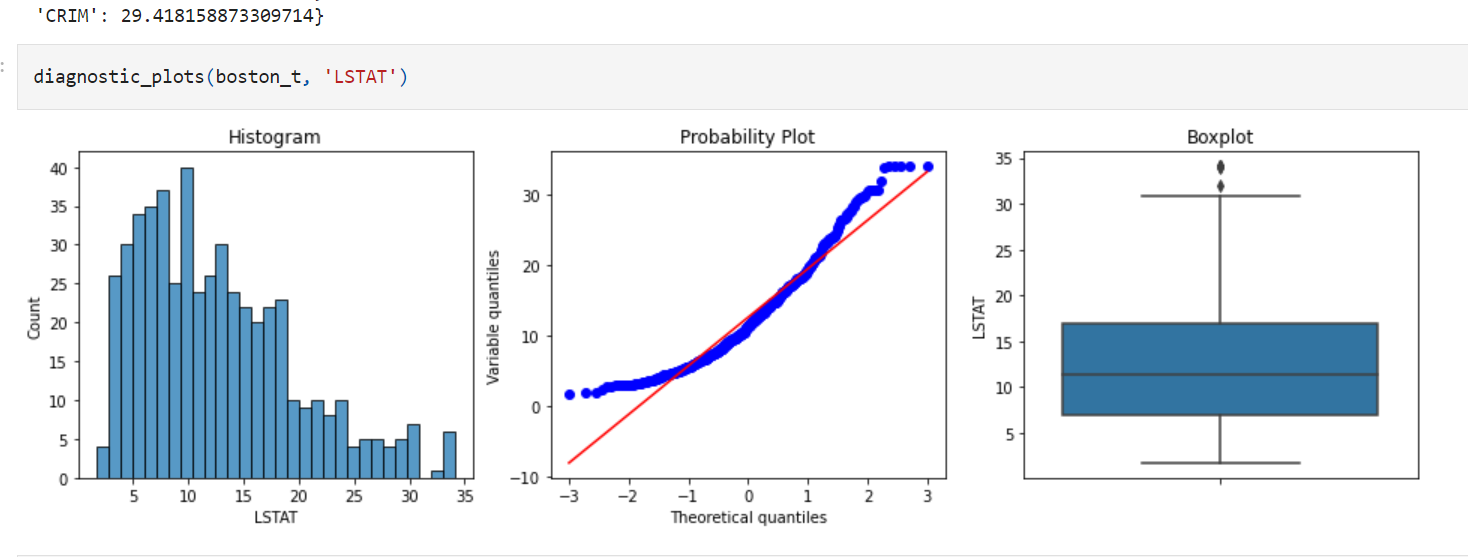












None of these variables are normally distributed, that is probably why the Gaussian approximation was not so effective to remove outliers. We could try and use a smaller distance, instead of multiplying by 3 times the std, we could 2 times or 1.5. But those numbers are set arbitrarily, and do not pose much statistical sense, therefore defeating the point of using the Gaussian approximation.

If this capping does not work as desired, I recommend you use the IQR rule as we discussed in the previous notebook, or quantiles, as we will see in the next one.

**Censoring or Capping.**

**Censoring**, or **capping**, means capping the maximum and /or minimum of a distribution at an arbitrary value. On other words, values bigger or smaller than the arbitrarily determined ones are **censored**.

Capping can be done at both tails, or just one of the tails, depending on the variable and the user.

Check my talk in [pydata](https://www.youtube.com/watch?v=KHGGlozsRtA) for an example of capping used in a finance company.

The numbers at which to cap the distribution can be determined:

* arbitrarily
* using the inter-quantal range proximity rule
* using the gaussian approximation
* using quantiles

**Advantages**

* does not remove data

**Limitations**

* distorts the distributions of the variables
* distorts the relationships among variables

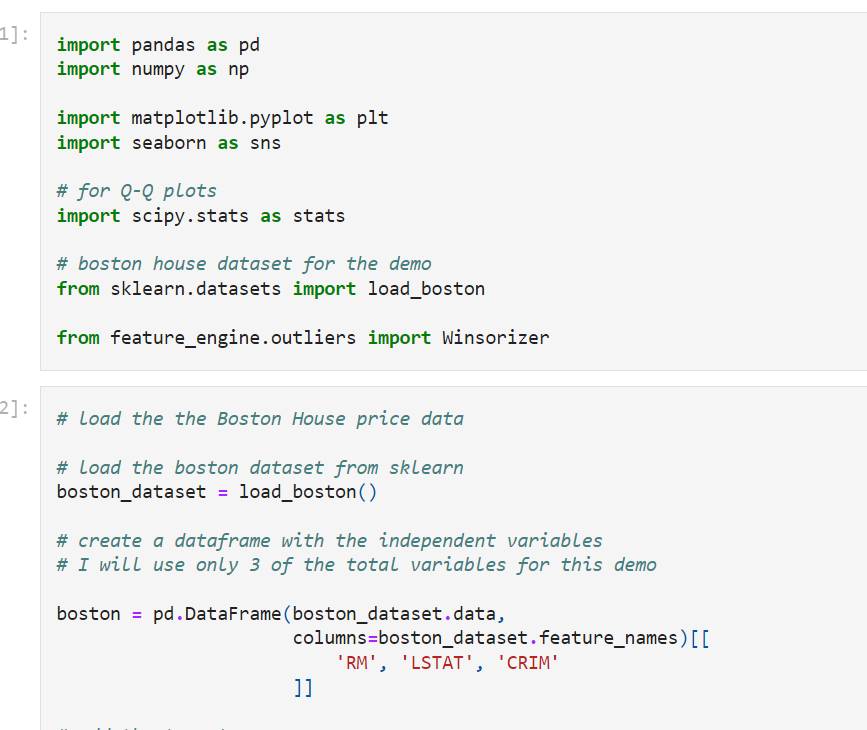
**In this Demo**

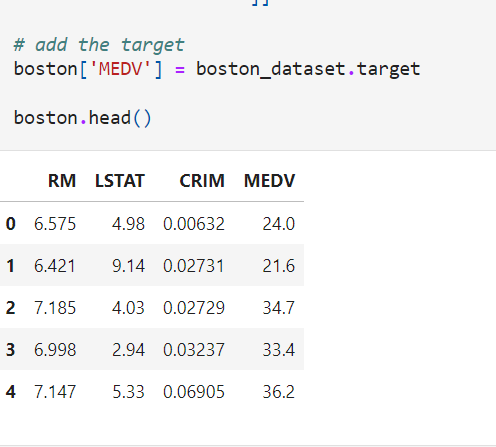
We will see how to perform capping with the quantiles using the Boston House Dataset

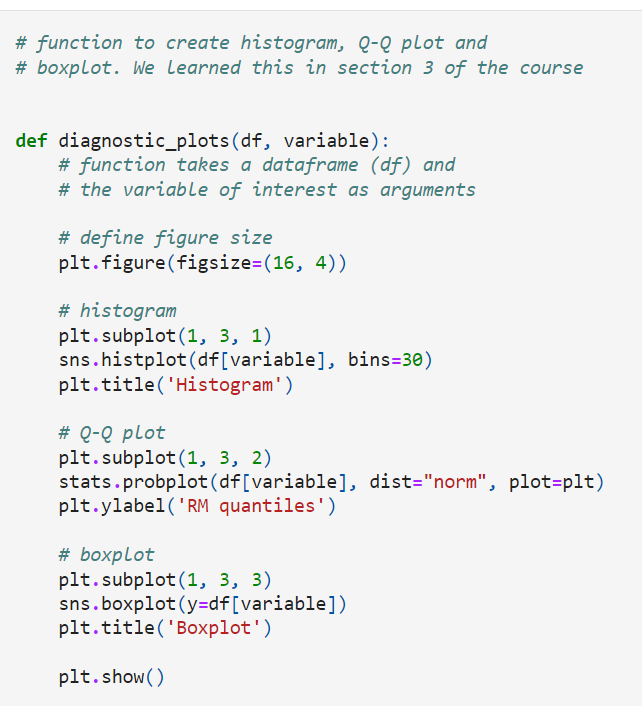
**Important**

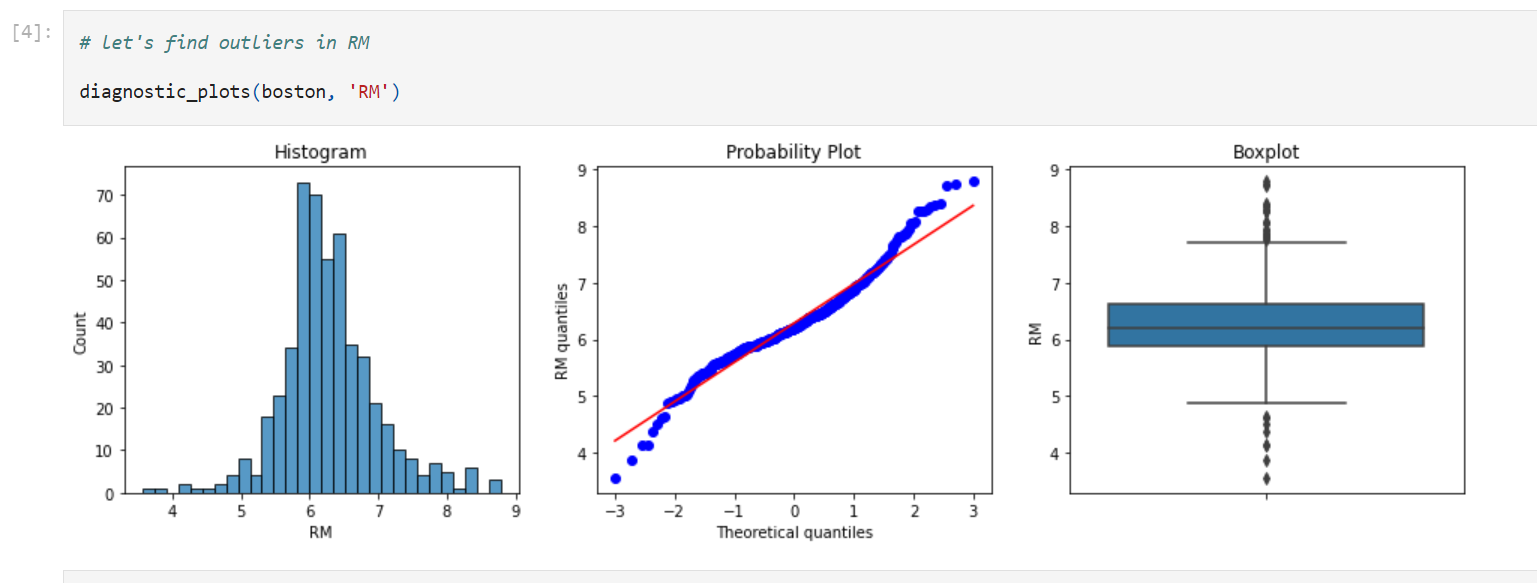
When doing capping, we tend to cap values both in train and test set. It is important to remember that the capping values MUST be derived from the train set. And then use those same values to cap the variables in the test set

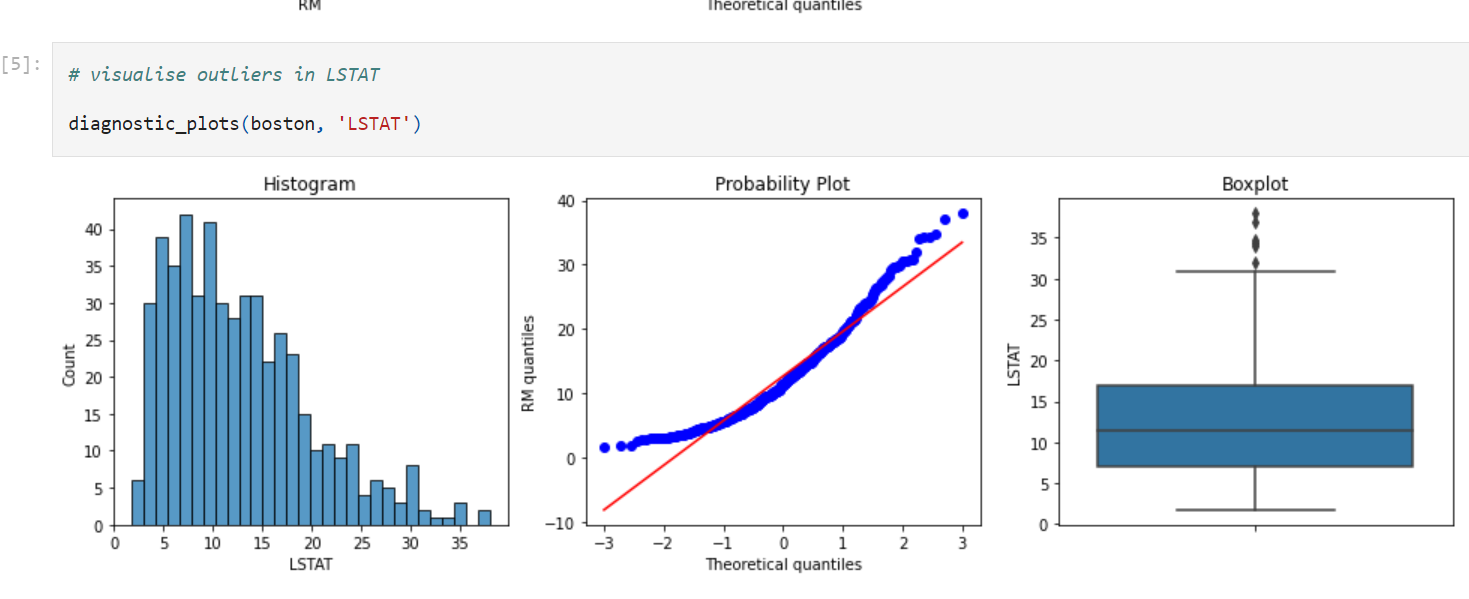
I will not do that in this demo, but please keep that in mind when setting up your pipelines

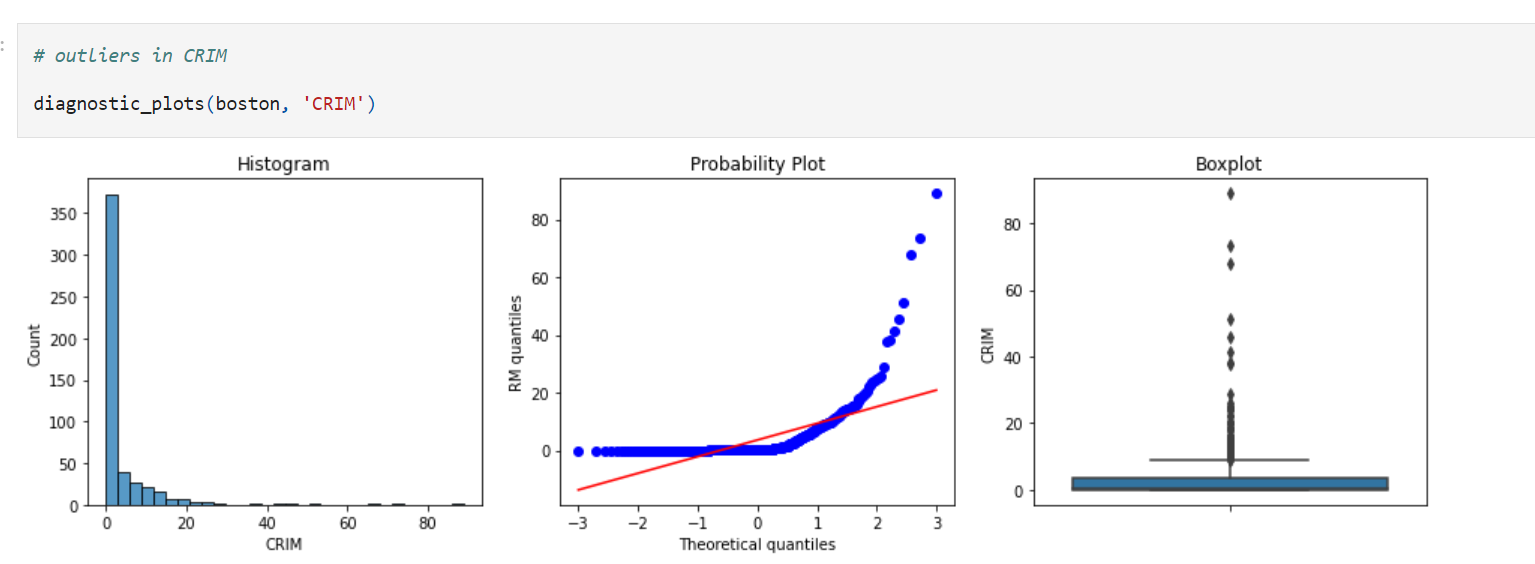






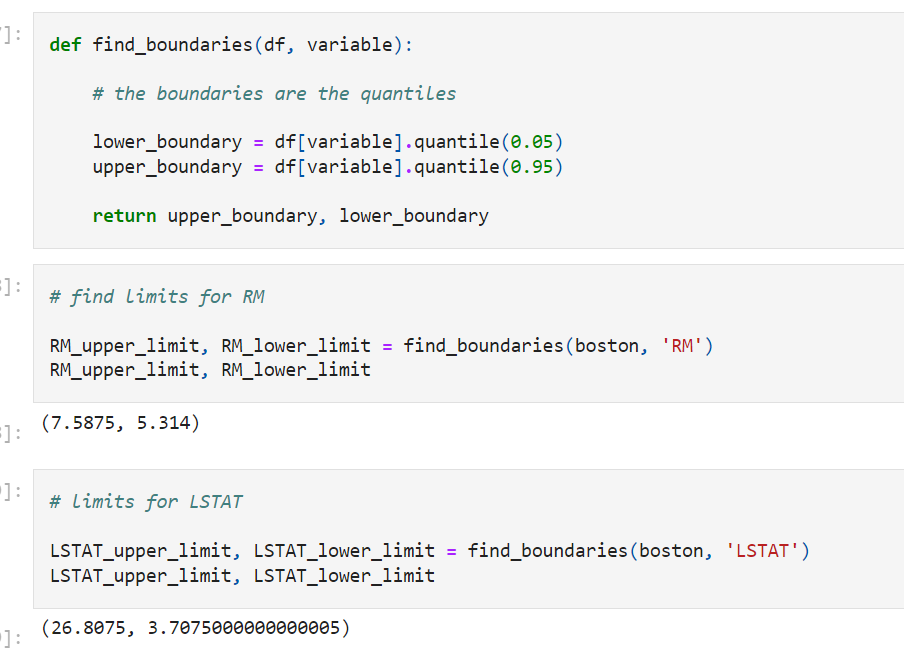


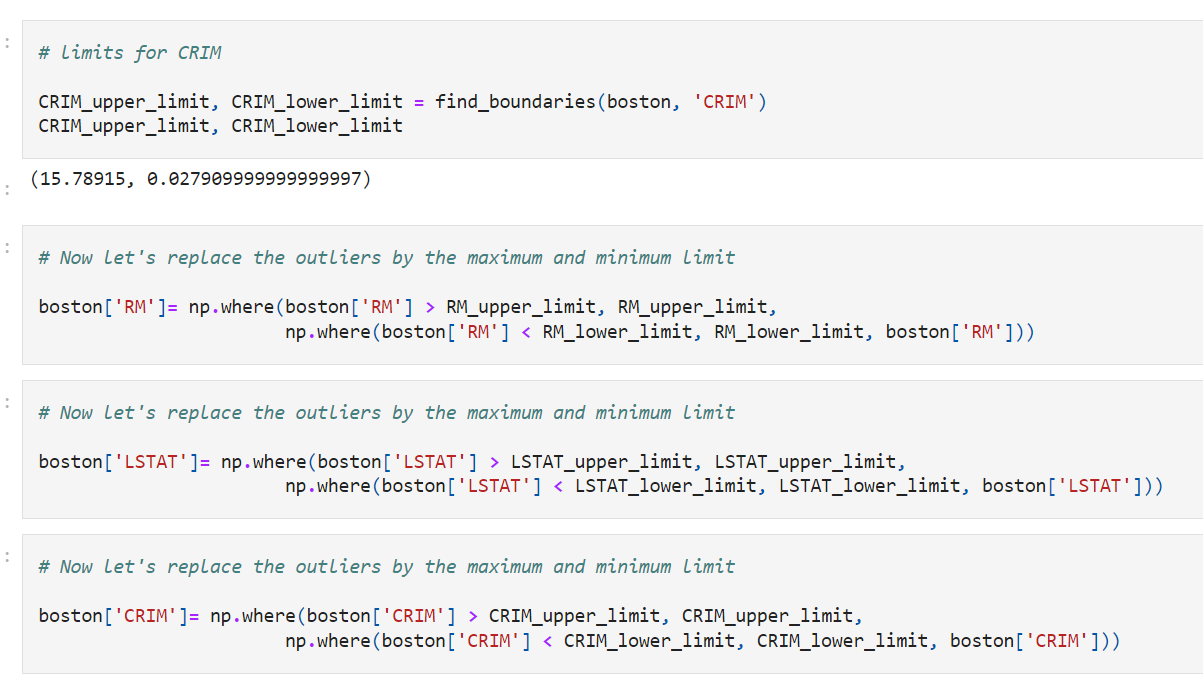


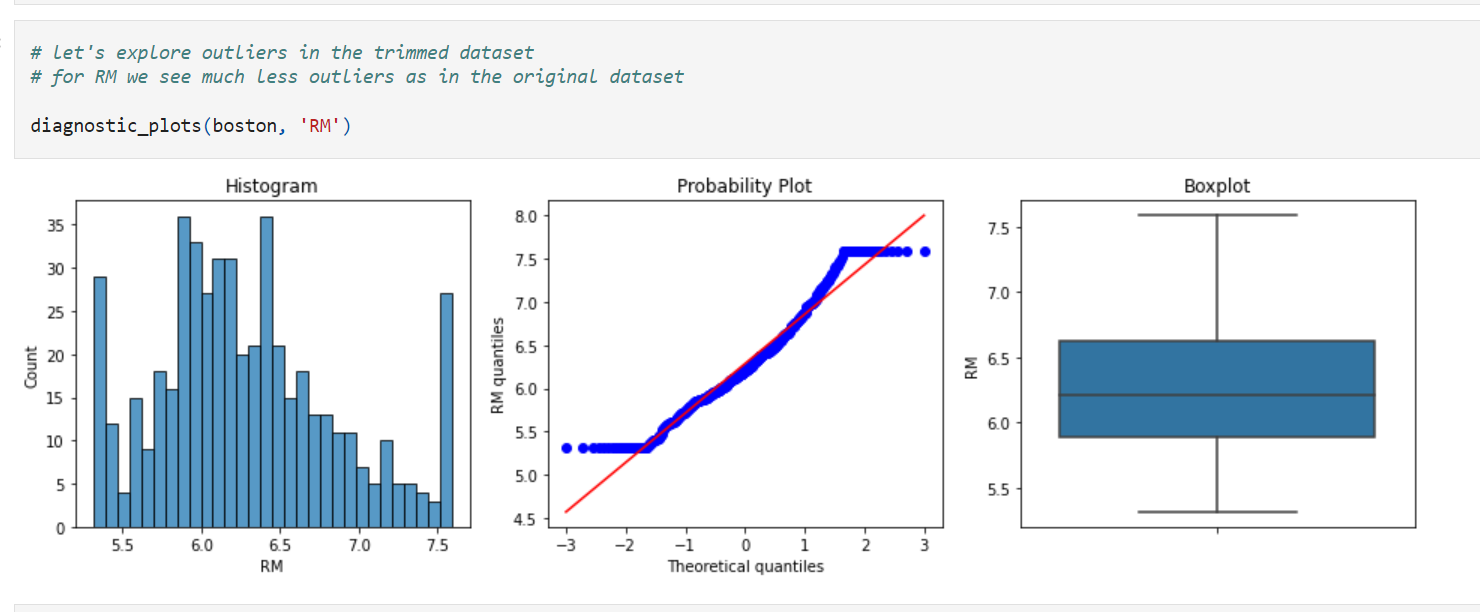


There are outliers in all of the above variables. RM shows outliers in both tails, whereas LSTAT and CRIM only on the right tail.

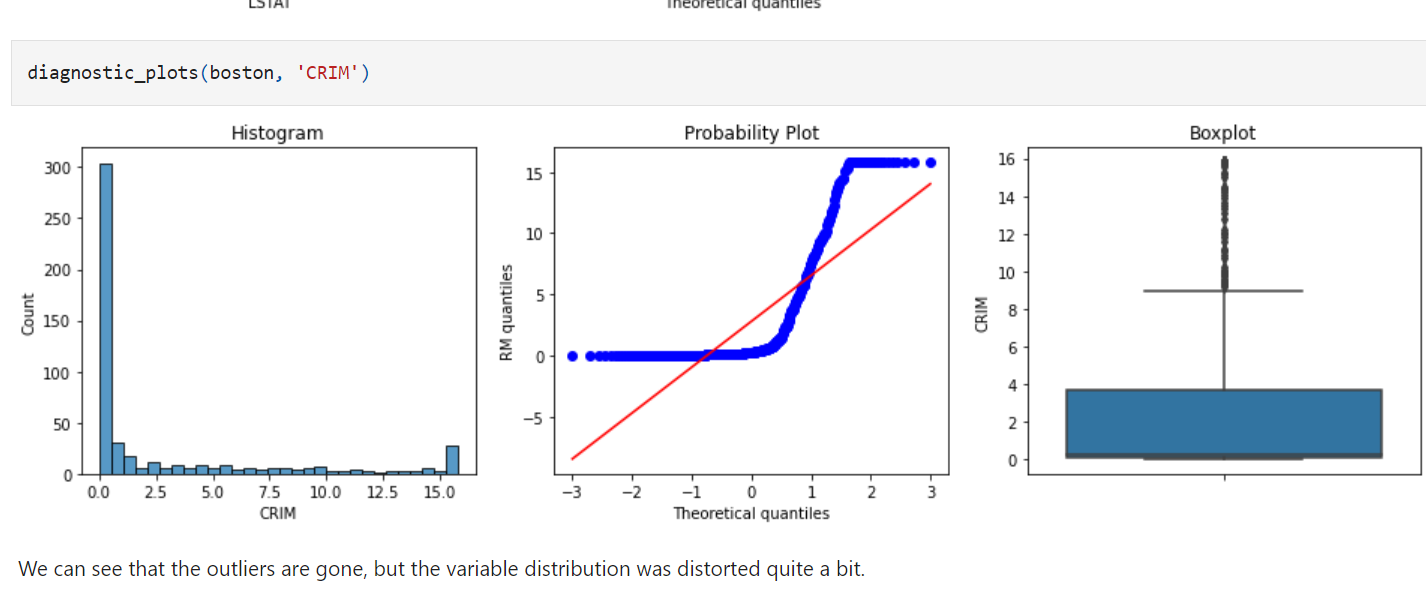
To find the outliers, let's re-utilise the function we learned in section 3:

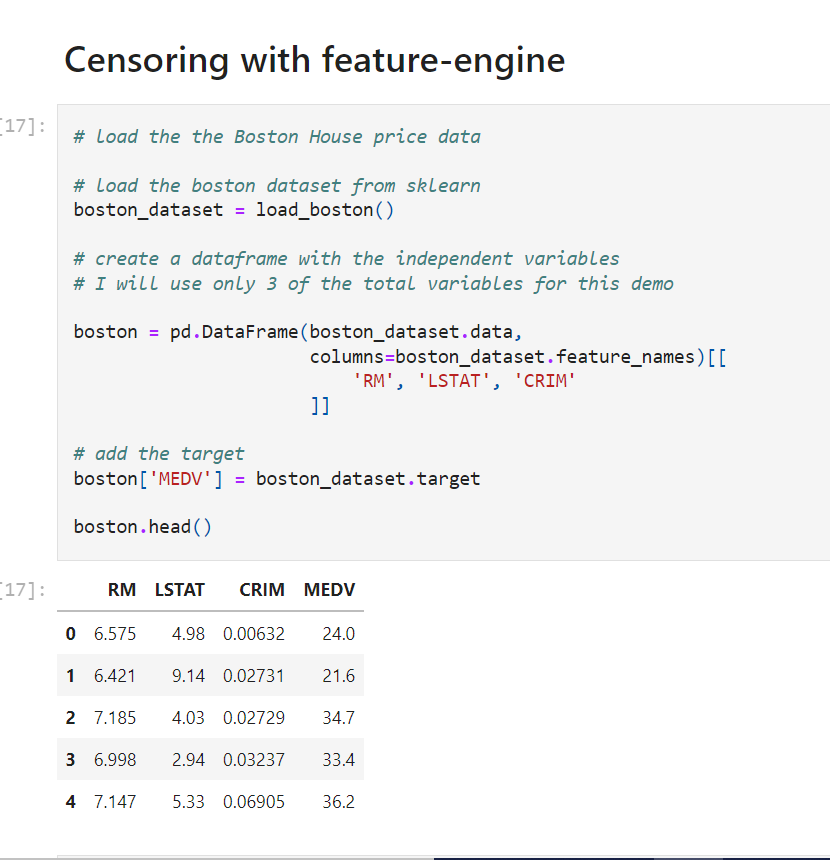


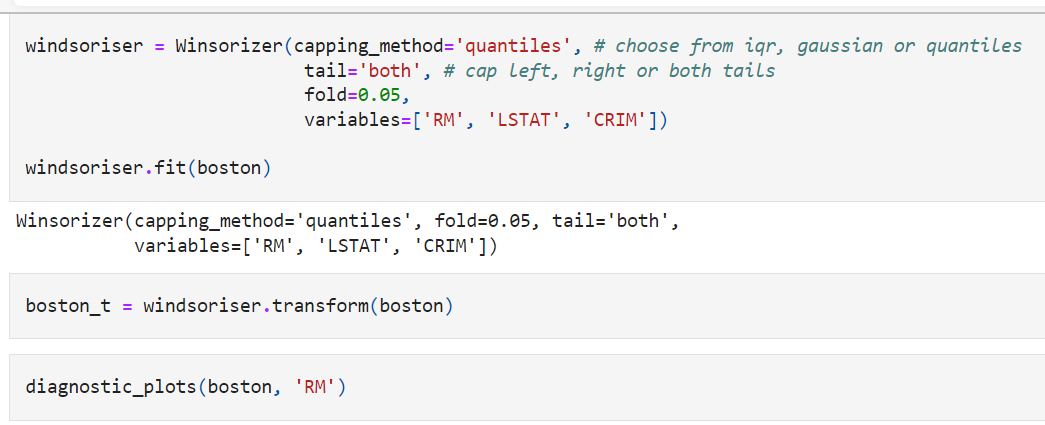


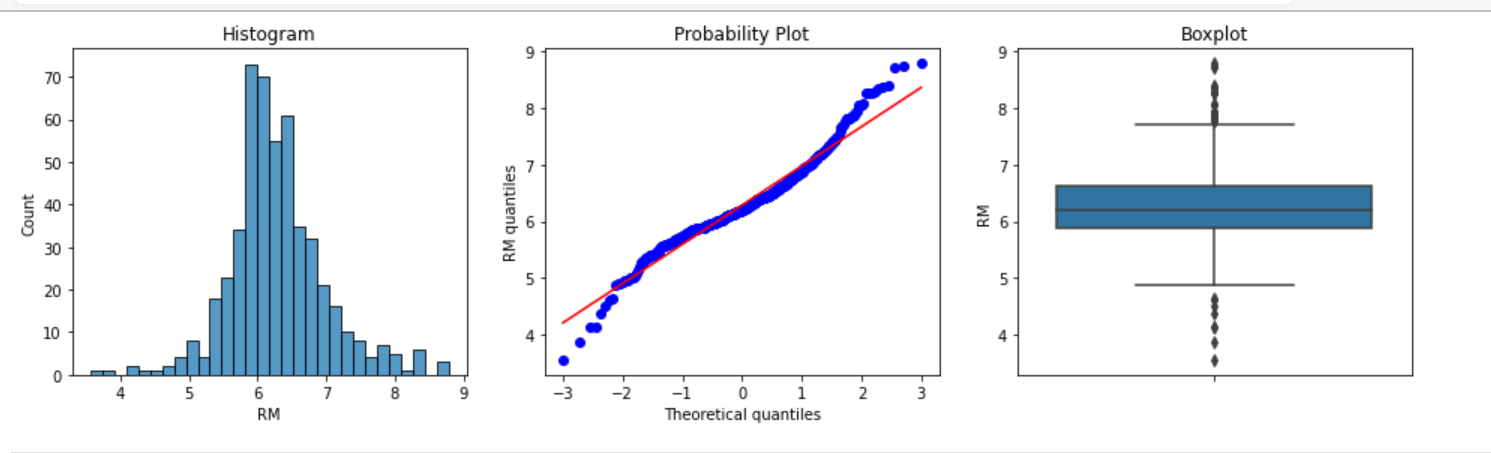


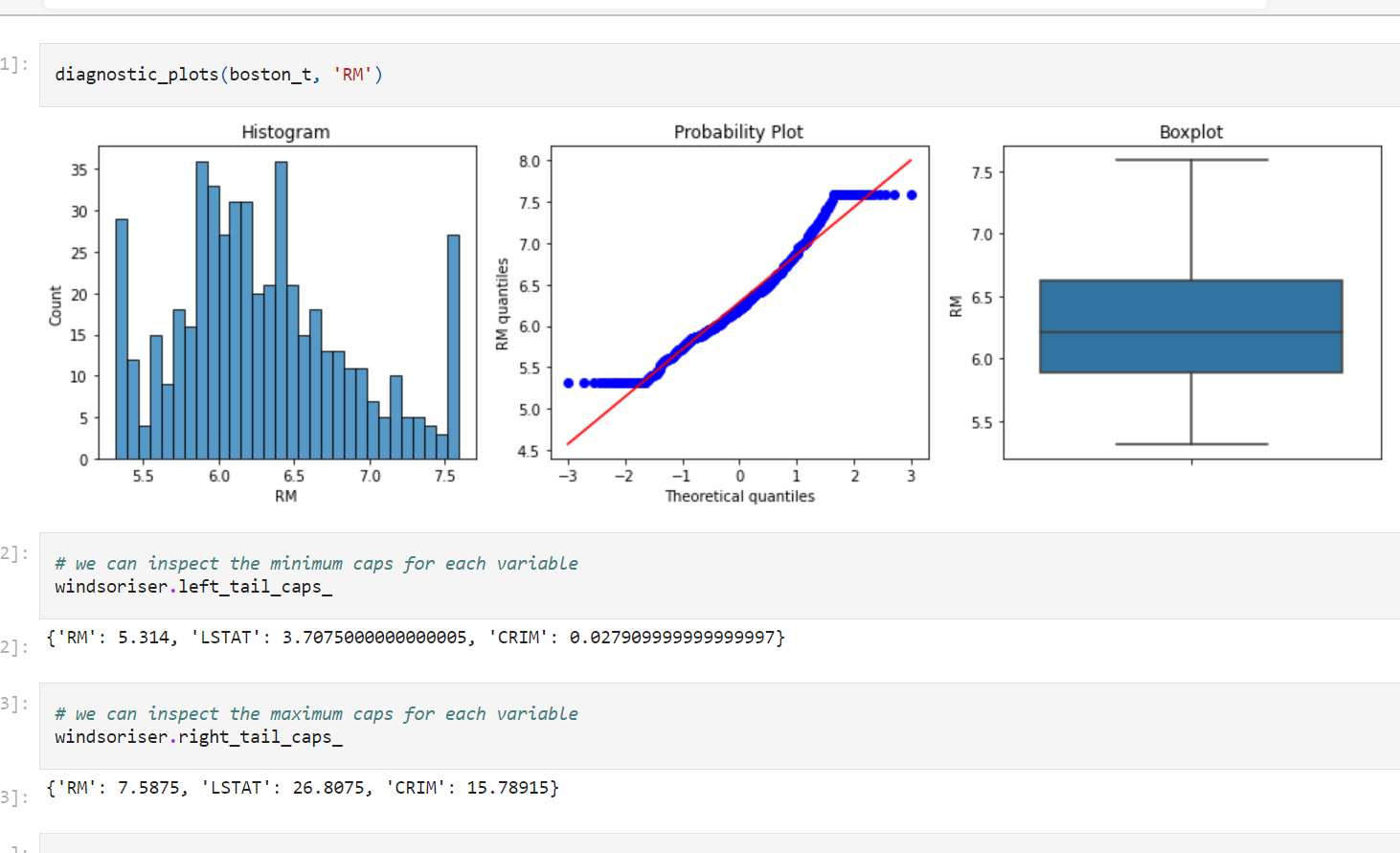












**Censoring or Capping.**

**Censoring**, or **capping**, means capping the maximum and /or minimum of a distribution at an arbitrary value. On other words, values bigger or smaller than the arbitrarily determined ones are **censored**.

Capping can be done at both tails, or just one of the tails, depending on the variable and the user.

Check my talk in [pydata](https://www.youtube.com/watch?v=KHGGlozsRtA) for an example of capping used in a finance company.

The numbers at which to cap the distribution can be determined:

* arbitrarily
* using the inter-quantal range proximity rule
* using the gaussian approximation
* using quantiles

**Advantages**

* does not remove data

**Limitations**

* distorts the distributions of the variables
* distorts the relationships among variables

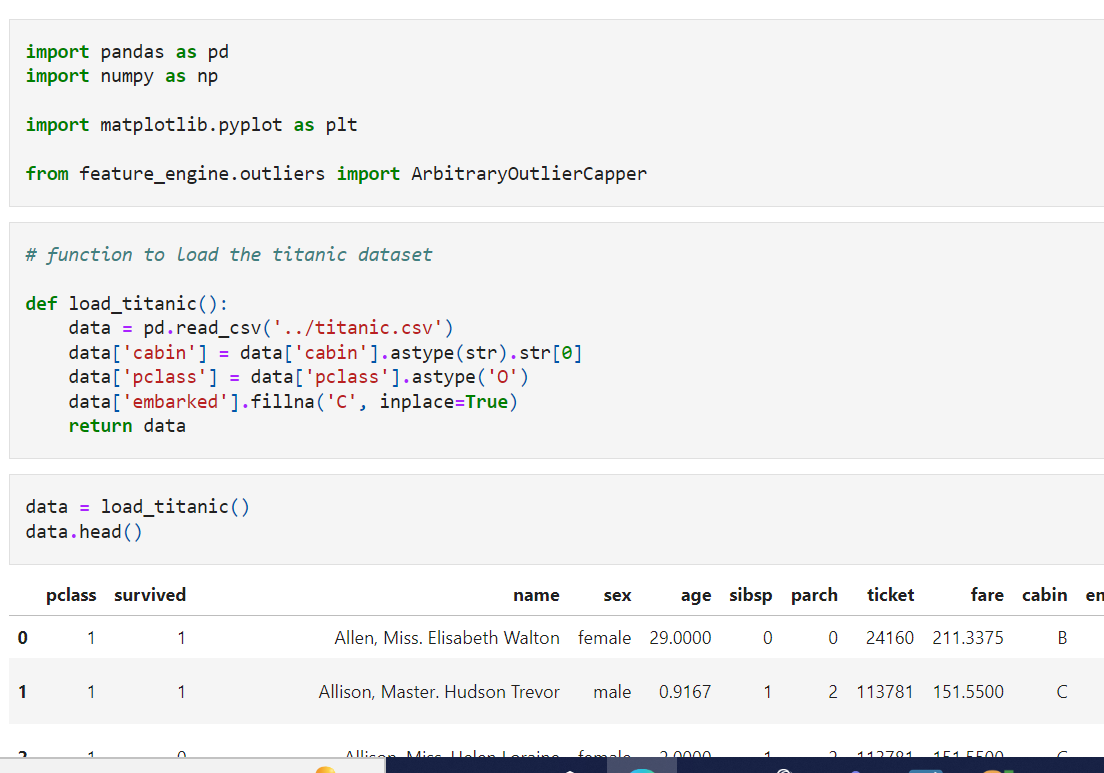
**In this Demo**

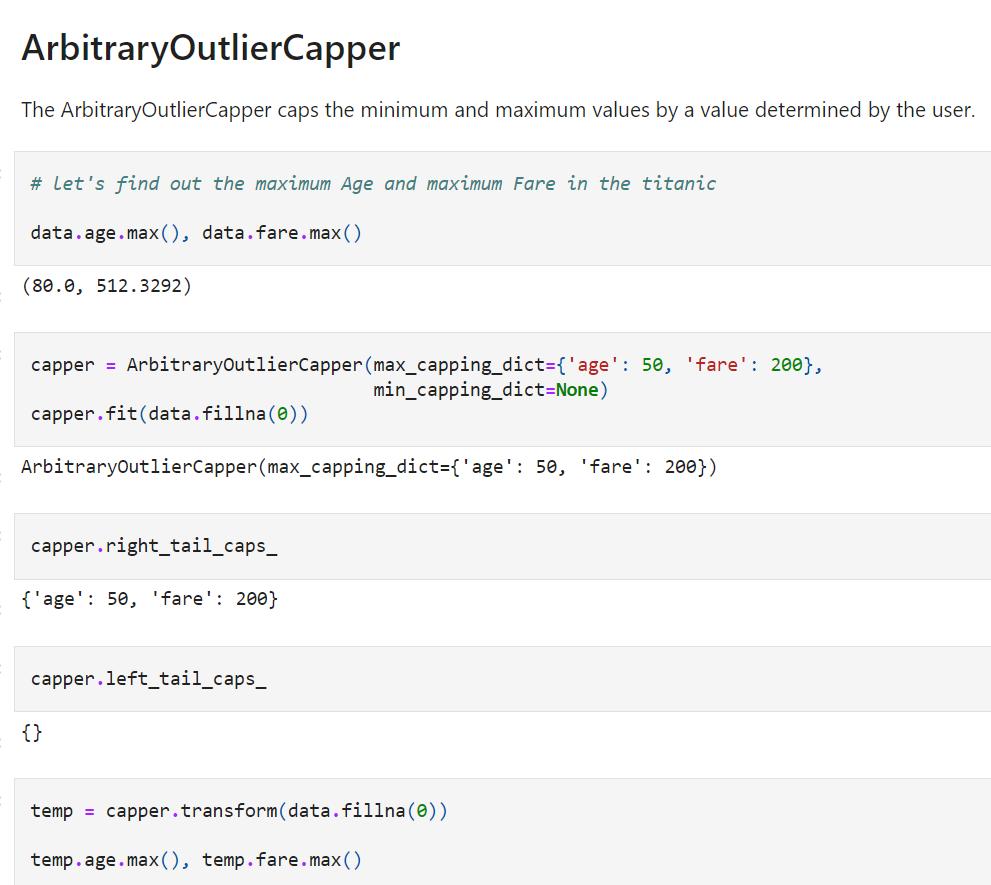
We will see how to perform capping with arbitrary values using the Titanic dataset

**Important**

When doing capping, we tend to cap values both in train and test set. It is important to remember that the capping values MUST be derived from the train set. And then use those same values to cap the variables in the test set

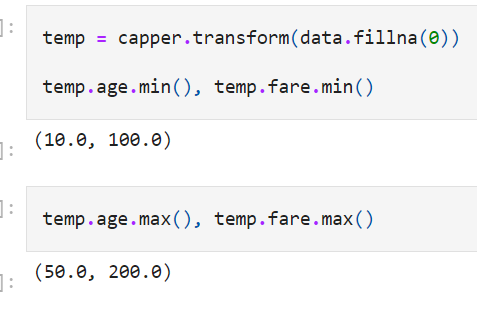
I will not do that in this demo, but please keep that in mind when setting up your pipelines











**Feature-engine**

We can also automate this process with Feature-engine: <https://feature-engine.readthedocs.io/en/latest/outliers/ArbitraryOutlierCapper.html>