Simple Imputation

Once you have decided to fill in missing values (and grappled with the implications of doing so) the simplest approach is to treat each column/feature separately and use some chosen value, such as the mean of the available data as the imputation value.

For example, using the data defined in the complete case analysis:

```
from sklearn.impute import SimpleImputer

features = ['price', 'inventory']
imp = SimpleImputer()

# Use .values attribute bc sklearn works with arrays rather than DataFrames
imp.fit(df[features].values)

print(imp.transform(df[features].values))

[[ 1.95  17.5  ]
 [ 3.    12.   ]
 [ 2.475 23.   ]]

```

As always, the best choice for exactly how and whether to impute missing data will depend on the nature of the data at hand and the overall goals of the project. You may need to try a few different options and compare how your model performs on some hold out data. Of course, at this point in the data science workflow you haven’t built a model yet, so you would just want to make a note to reconsider your imputation methods once you get to the back-and-forth between the feature engineering and modeling phases of the project. This sort of back-and-forth also underlies the big picture approach to missing values taken with multiple imputation.