1. **INTERPRETING ONE-HOT ENCODED COEFFICIENTS**

That is a much more manageable number of coefficients. Let's go through and interpret these:

\* The **\*\*reference category\*\*** for `origin` is `1` (US) and for `make` is `amc` (American Motor Company)

\* `const`, `weight`, and `model year` are all still statistically significant

  \* When all other predictors are 0, the MPG would be about -18.3

  \* For each increase of 1 lb in weight, we see an associated decrease of about 0.006 in MPG

  \* For each year newer the vehicle is, we see an associated increase of about 0.75 in MPG

\* `origin\_2` and `origin\_3` are not statistically significant any more

  \* While this might seem surprising, our data understanding can explain it. The `origin` feature and the `make` feature are really providing the same information, except that `make` is more granular. Every `make` category (except for `other`) corresponds to exactly one `origin` category. Therefore it probably does not make sense to include both `origin` and `make` in the same model

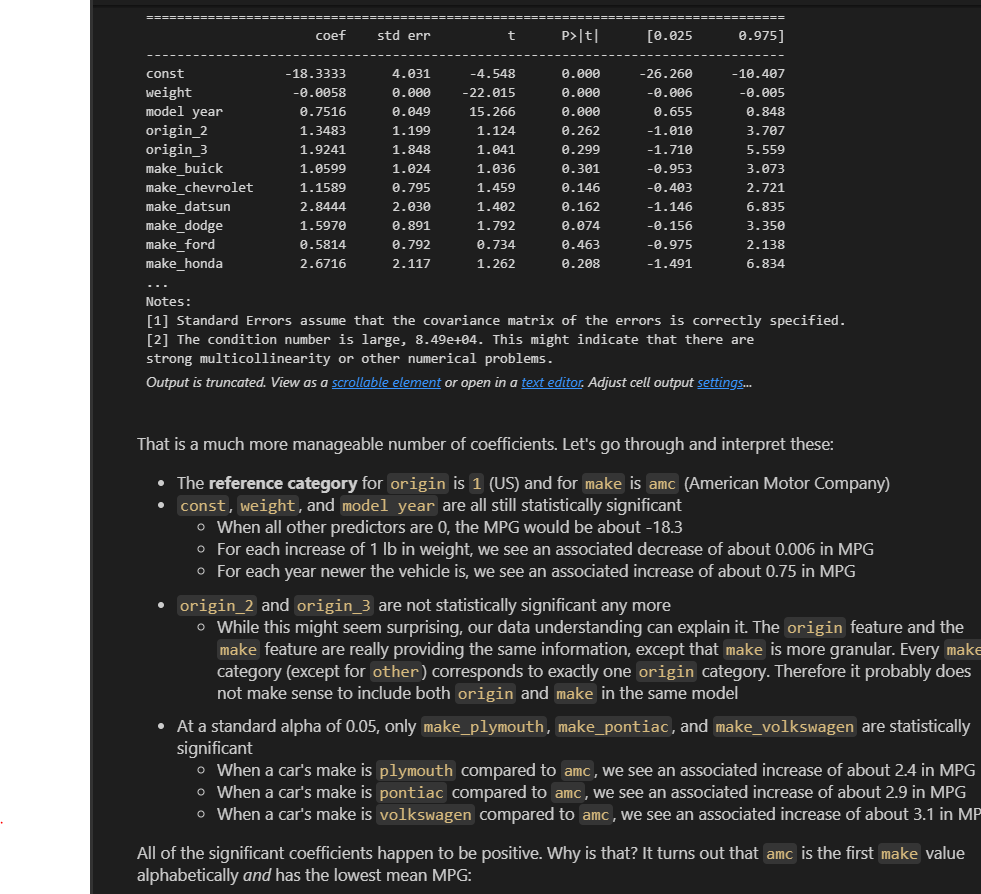
\* At a standard alpha of 0.05, only `make\_plymouth`, `make\_pontiac`, and `make\_volkswagen` are statistically significant

  \* When a car's make is `plymouth` compared to `amc`, we see an associated increase of about 2.4 in MPG

  \* When a car's make is `pontiac` compared to `amc`, we see an associated increase of about 2.9 in MPG

  \* When a car's make is `volkswagen` compared to `amc`, we see an associated increase of about 3.1 in MPG

All of the significant coefficients happen to be positive. Why is that? It turns out that `amc` is the first `make` value alphabetically *\_and\_* has the lowest mean MPG:



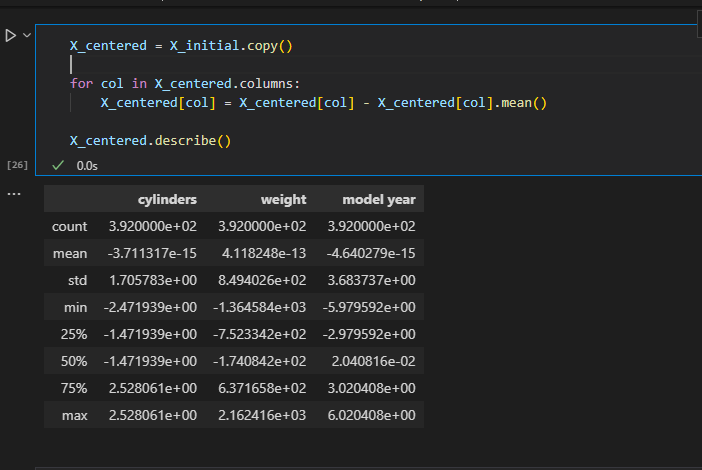
1. **CENTERING**

X\_centered = X\_initial.copy()

for col in X\_centered.columns:

    X\_centered[col] = X\_centered[col] - X\_centered[col].mean()

X\_centered.describe()



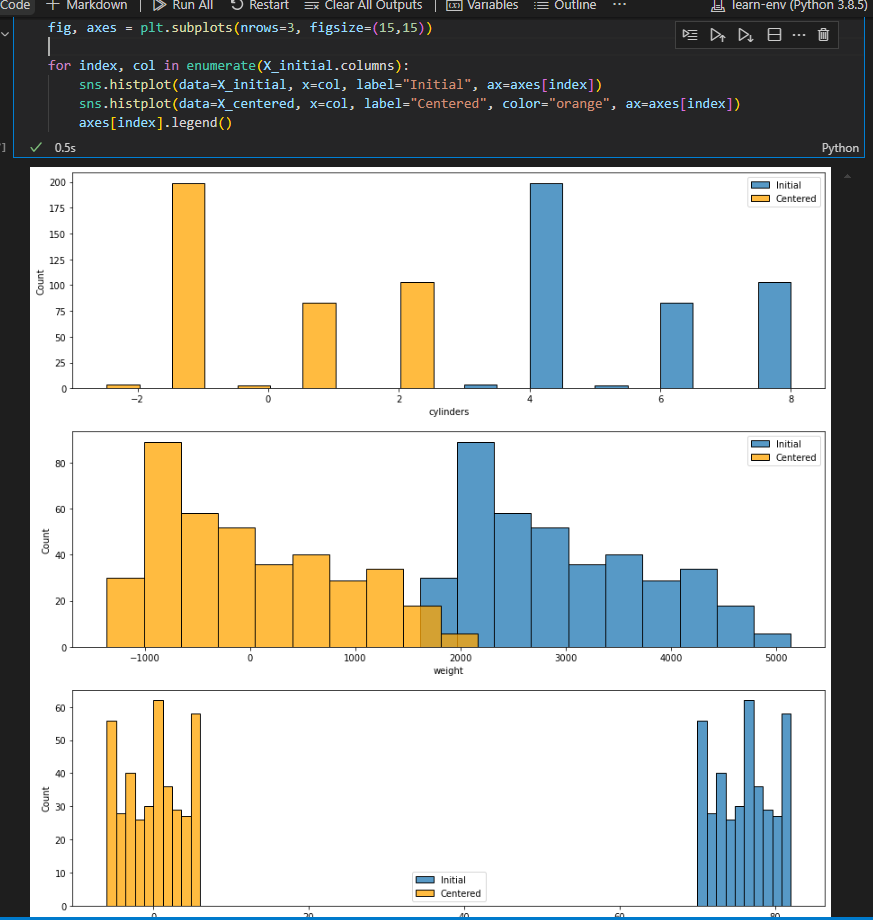
fig, axes = plt.subplots(nrows=3, figsize=(15,15))

for index, col in enumerate(X\_initial.columns):

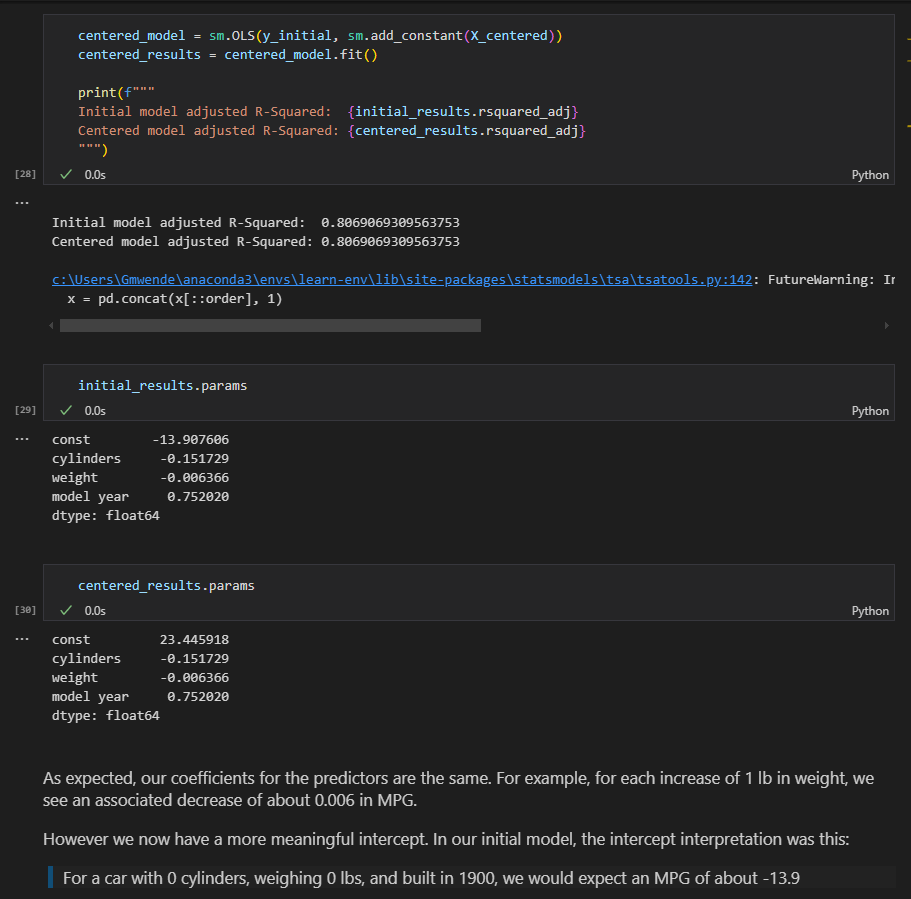
    sns.histplot(data=X\_initial, x=col, label="Initial", ax=axes[index])

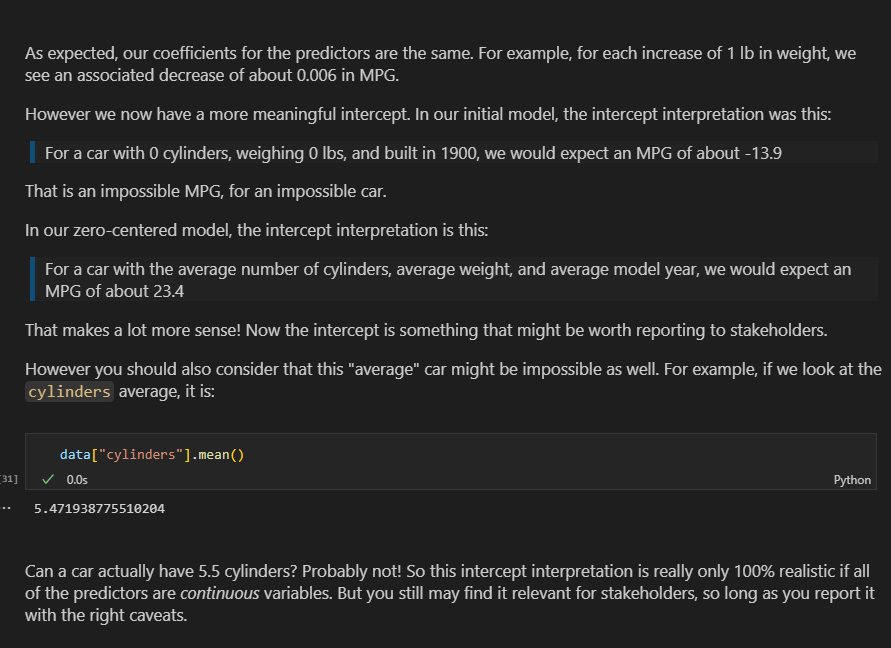
    sns.histplot(data=X\_centered, x=col, label="Centered", color="orange", ax=axes[index])

    axes[index].legend()

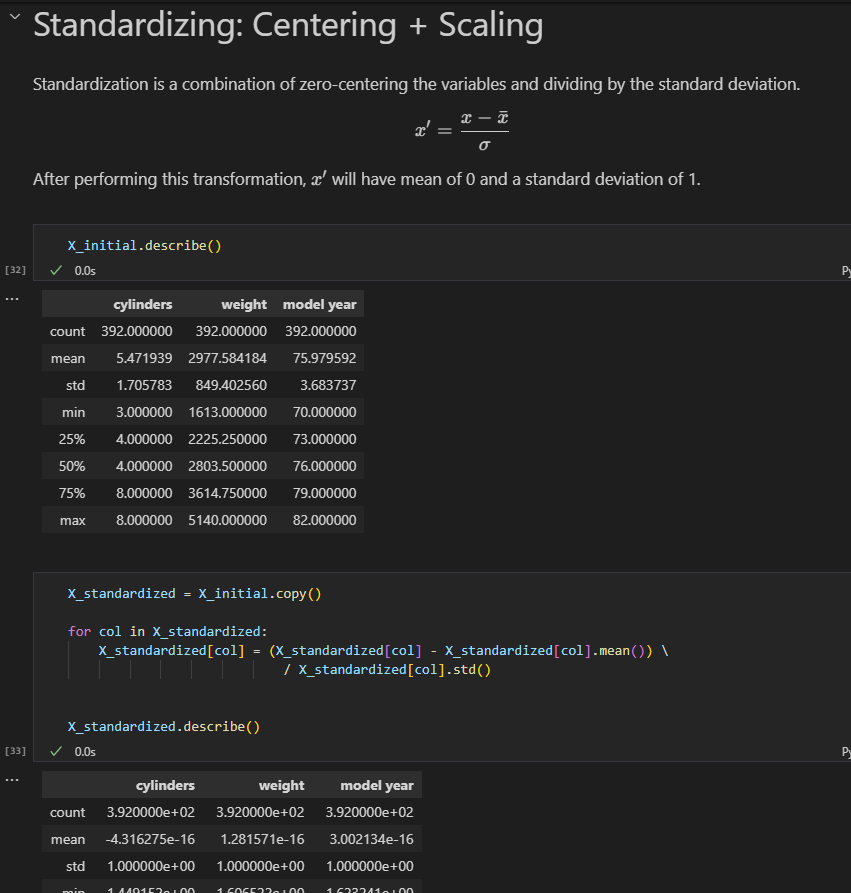


On modelling we can now our coefficients are interpretable

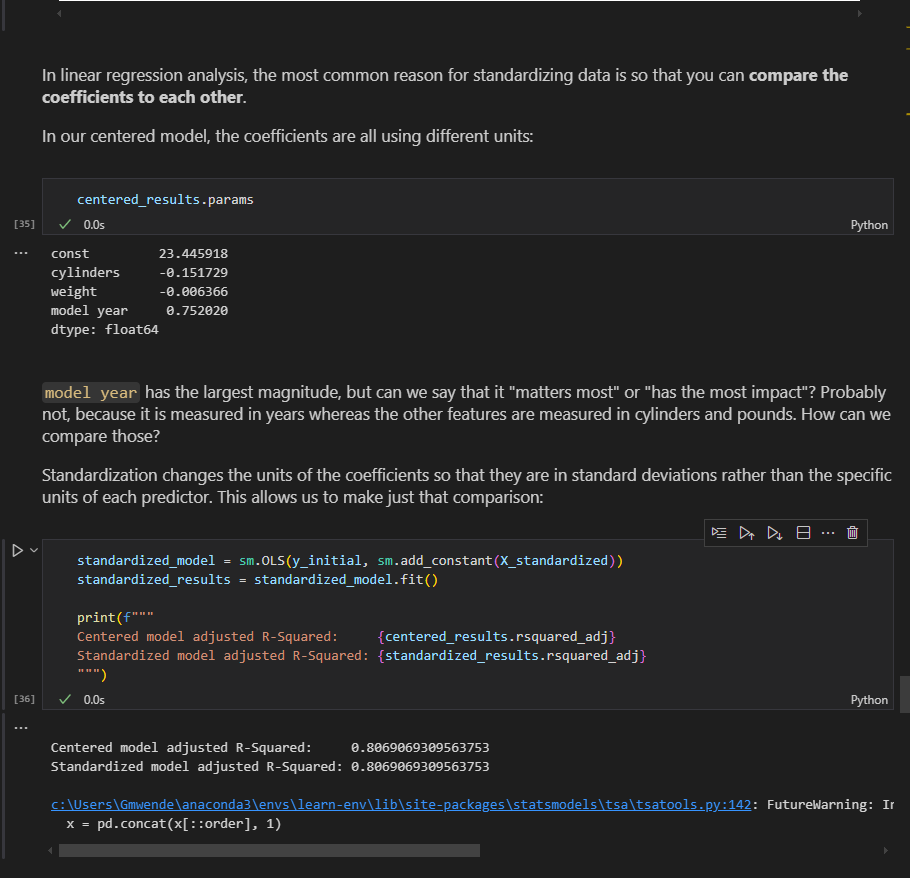


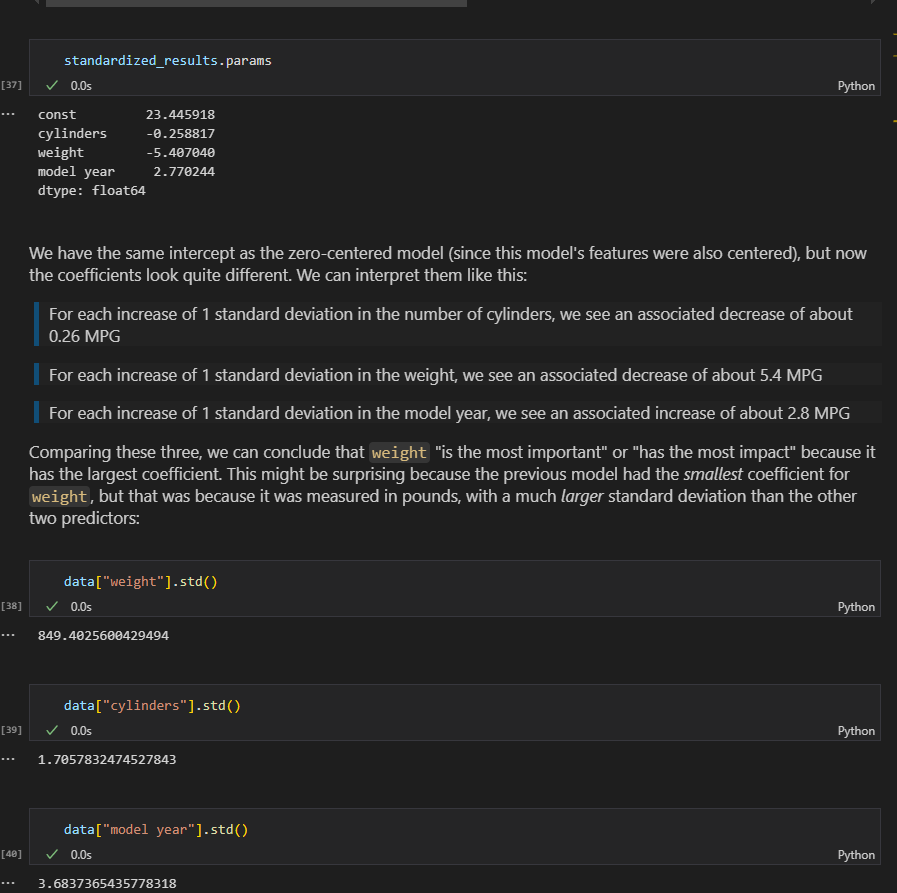


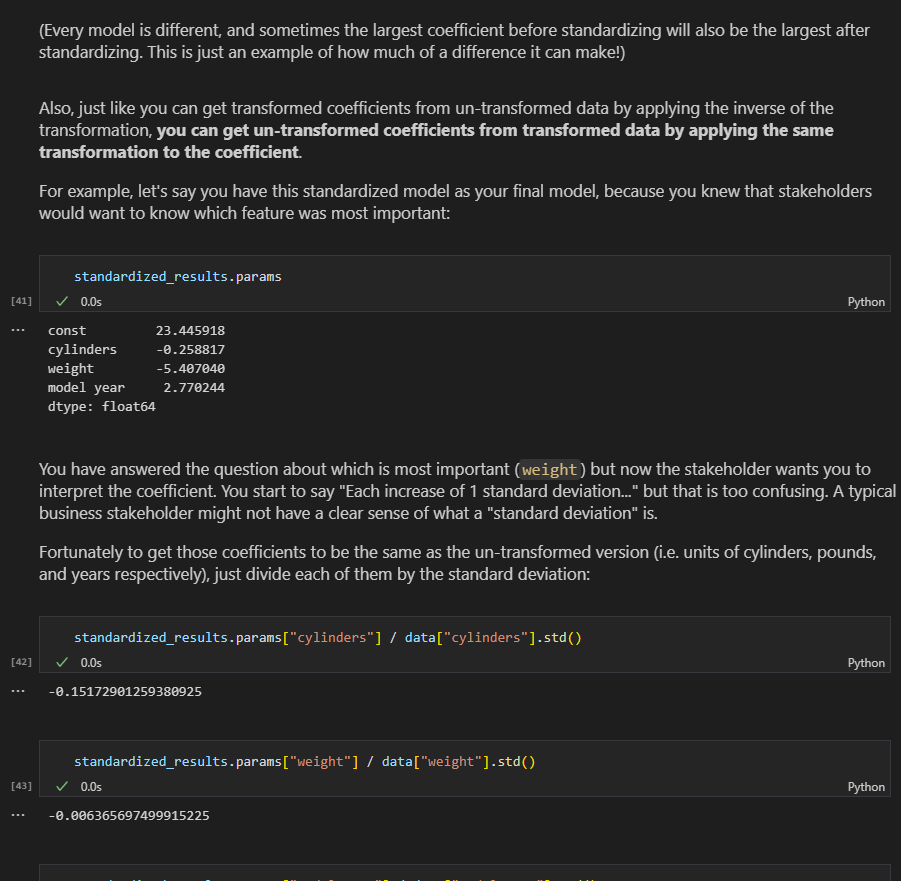
1. Standardization

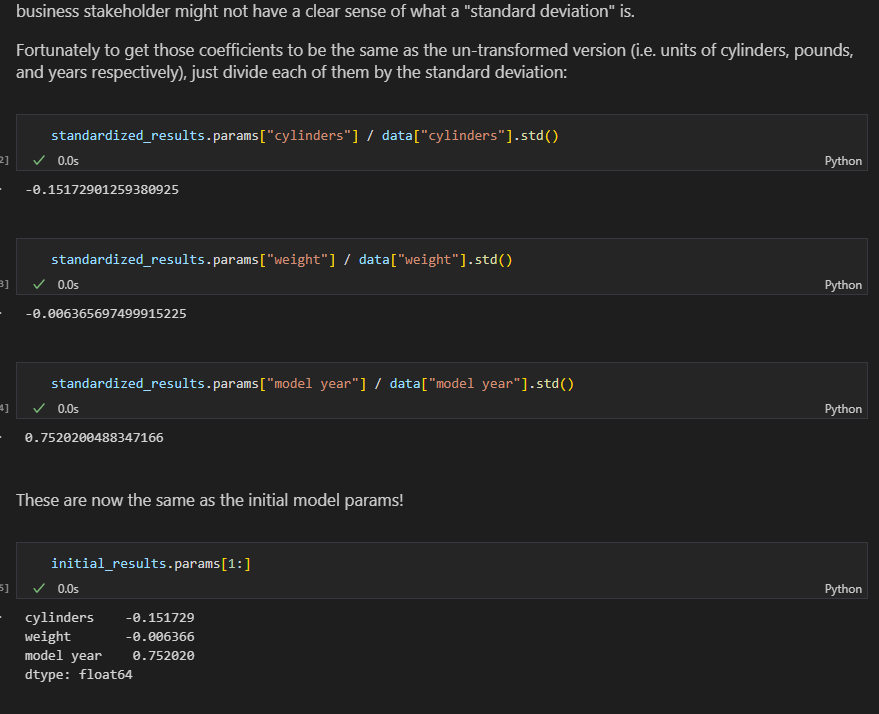












Standardization using sklearn

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

x\_sk\_standarized = scaler.fit\_transform(X\_initial)

standardized\_model2 = sm.OLS(y\_initial, sm.add\_constant(x\_sk\_standarized))

standardized\_results2 = standardized\_model2.fit()

standardized\_results2.summary()

