**Preprocessing of continuous variables**

There are two continuous variables: macro\_state\_1 and macro\_state\_2. Median replacement was implemented to handle the outliers (mainly in macro\_state\_1).

We first checked their individual impact on the outcomes, with consideration of up to 3rd order polynomial models. This is achieved by **“preprocess\_macro\_state.py”.**

See XY scatter plots for Y-outcome and X- macro\_state\_1 (or 2) (left column), and for Y-outcome vs X-transformed polynomial feature (right column)

*(In these plots, the category variable is label in different colors)*

|  |  |  |
| --- | --- | --- |
| **Model order** | **Outcome vs. macro\_state\_1** | **Outcome vs. feature (polynomial)** |
| 1st |  | A diagram of a graph  Description automatically generated |
| 2nd |  | A graph of a graph showing a line of different colored dots  Description automatically generated with medium confidence |
| 3rd |  | A diagram of a graph  Description automatically generated |

The difference between 2nd and 3rd order polynomials is small, suggesting that 2nd order polynomial is sufficient as a feature to explain the variance of Y-outcome.

|  |  |  |
| --- | --- | --- |
| **Model order** | **Outcome vs. macro\_state\_2** | **Outcome vs. feature (polynomial)** |
| 1st | A graph of a graph showing a number of different colored dots  Description automatically generated with medium confidence | A graph of a graph showing a line of different colored dots  Description automatically generated with medium confidence |
| 2nd | A diagram of a graph  Description automatically generated | A graph of a graph showing a line of different colored dots  Description automatically generated with medium confidence |
| 3rd | A diagram of a graph  Description automatically generated | A graph of a graph showing a line of different colored dots  Description automatically generated with medium confidence |

Similarly, for macro\_state\_2, 2nd order polynomial (as a transformed feature) is sufficient as a feature to explain the variance of Y-outcome.

So the final model can be expressed as:

Y(outcome) = a + b1 \* X1 + b2\* (X1)2 + c1 \* X2 + c2\* (X2)2 + other categorical terms

where X1 is macro\_state\_1, and X2 is macro\_state\_2.

The feature 1 and 2 for the linear model are: b1 \* X1 + b2\* (X1)2 and c1 \* X2 + c2\* (X2)2

**Selection of categorical variables for final model**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Average**  **R2** | **Intercept** | **macro\_state\_1**  **1st order coef.** | **macro\_state\_1**  **2nd order coef.** | **macro\_state\_2**  **1st order coef.** | **macro\_state\_2**  **2nd order coef.** | **Innovation**  **success** | **category** | **year** |
| 0.952 | 161.617 | -161.168 | 25.033 | 149.658 | -25.174 | 19.859 | 41.405 | 10.322 |
| 0.951 | 190.741 | -167.908 | 25.891 | 149.756 | -25.183 | 19.895 | 41.468 |  |
| 0.948 | 321.145 | -161.452 | 25.045 | 149.304 | -25.133 |  | 41.504 | 10.582 |
| 0.948 | 351.302 | -168.362 | 25.925 | 149.403 | -25.142 |  | 41.569 |  |
| 0.945 | 244.165 | -161.418 | 25.021 | 149.734 | -25.187 | 19.943 |  | 10.712 |
| 0.945 | 274.522 | -168.413 | 25.912 | 149.835 | -25.197 | 19.981 |  |  |
| 0.941 | 404.572 | -161.703 | 25.033 | 149.378 | -25.146 |  |  | 10.974 |
| 0.941 | 435.983 | -168.870 | 25.946 | 149.481 | -25.156 |  |  | NA |

Now we evaluate the R2 values when categorical variables (“category”, “innovation\_success”, and “year”) are included. All combinations are considered, and a 10-fold cross-validation scheme was used. The averaged R2 value for all ten folds is reported as the final model ranking metric. This is achieved by “train\_model.py”.

From the summary above, when all three categorical variables are used, the R2 is highest, but to a negligible amount of 0.001 (compared to the model without using “year”).

To use a model with less variables but still keep a good performance, the model with only “category” appears to be the best choice, with a R2 value of 0.948.

Therefore, for the hold-out test sample, we shall use the following model:

**Y = 351.302 + Feature1 +Feature2 + 41.569 \* Category**

Feature 1 = -168.362 \* X1 + 25.925(X1)2

Feature 2 = 25.925 \* X2 +149.403 (X2)2

This is equivalently to the polynomial model below:

Y = 351.302 + [-168.362 \* X1 + 25.925(X1)2] +[25.925 \* X2 +149.403 (X2)2] + 41.569 \* Category