# Linear Regression Experiment
# – Prediction on Institution Scores


## Abstract

The goal of the experiment is to predict institutions' scores with linear regression on the given data from Kaggle.

After discrete string feature values and filling in missing data, we run four linear regression models (OLS, Elastic, Lasso, Ridge). Neither of the models carry out preferable results with high R-Square and low RMSE(MSE).

Thus, further data analysis and processing is tested:

1. VIF select features
2. Lasso select features
3. Feature expansions
4. Remove outlying observations

Among the above, feature expansion(3) performs best with an expansion degree at 3; The evaluation scores under this expanded data model are around:
*MSE: 6.069026242066872*
*RMSE: 2.4635393729483748*
*R-Squared: 0.8991647525009168*

**This also suggests that the relationship between independent features and the dependent feature do not entirely follow a linear regression model as:** $Y = X\beta + \epsilon$ **;**
**But more like:** $Y = X^3\beta + \epsilon$ *or* $Y = X^4\beta + \varepsilon(X \text{ is the demision reduction of original } X)$ **.**


## 1 Introduction

University ranking is a challenging and controversial topic. At present, there are hundreds of evaluation agencies around the world that evaluate the comprehensive scores of universities for ranking, and the scores from these agencies are often inconsistent.
Among these agencies, Center for World University Rankings(CWRU) is one of the most influential organizations. It evaluates the quality of education, alumni employment, research results and references, rather than relying on data submitted by universities
In this task, we are going to construct a linear regression model to predict the comprehensive scores of universities.

## 2 Programming Modules and Techniques Used

In statistics, linear regression is a linear approach for modeling the relationship between a scalar response and one or more explanatory variables (also known as dependent and independent variables)[1].

Sklearn and statsmodels modules are constantly used in this experiment for constructing linear regression models and analyzing data.
Missingno, seaborn and matplotlib are invoked for data visualization.

## 3 Data Analyze and Pre-process

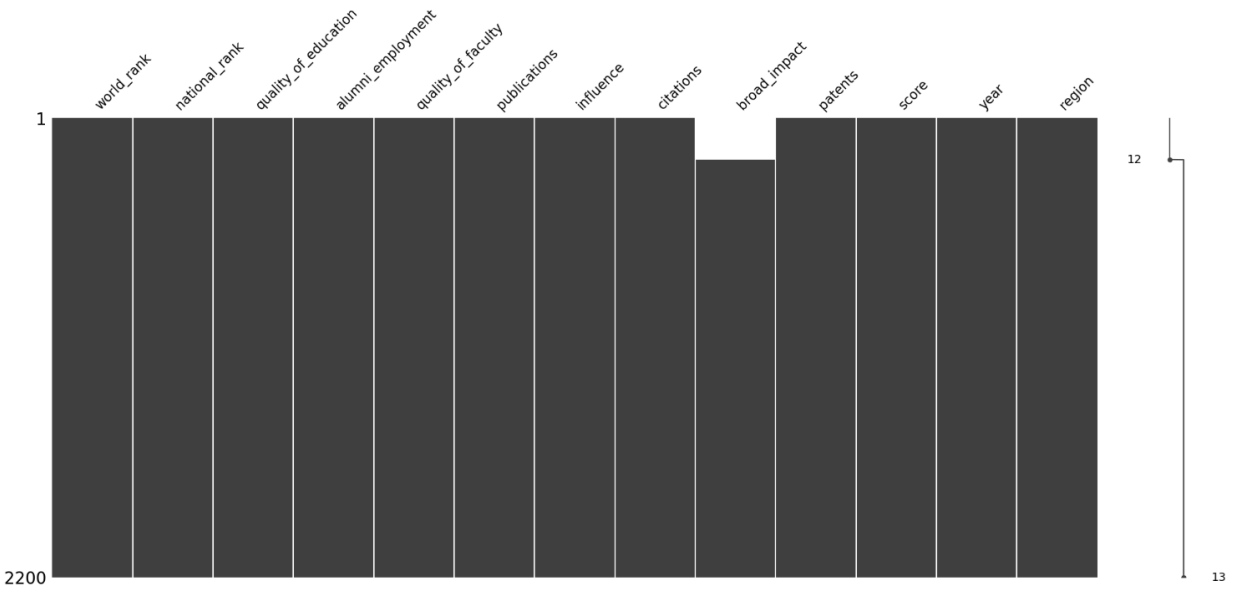
### 3-1. Discrete Region Category

Among the independent features, we noticed that there are two features: institution and region, whose values are in string type, not numeric type. Since the institution feature only describes the name of the university, we will drop it out from the data set.
The label encoder is used to convert categorical string data into numeric[2].

```
label_encoder = LabelEncoder()
labels = label_encoder.fit_transform(data_frame["region"])
data_frame.drop(columns="region", inplace=True)
data_frame["region"] = labels
```

### 3-2. Missing Data

Finding missing data is a standardized step to evaluate a data set. The handling of missing data is important during the preprocessing of the dataset as many machine learning algorithms do not support missing values[3].
Missingno is used to visualize the data missing in the current data set.



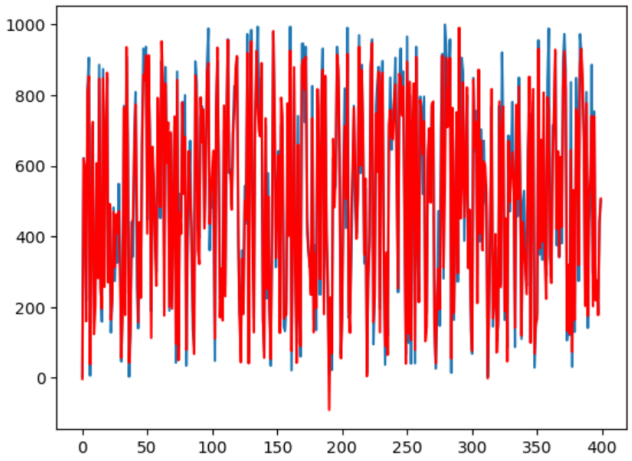

The plot implies that all missing data happens only in the independent feature “broad_impact”.

The easiest way to handle missing data is to delete rows that have such an occasion. The model trained with the missing data removed can be robust. But this still means loss of information. Thus, in our experiment, we will be looking at other statistical ways to fill in the missing ones.

### 3-2-1. Attempt to fill missing date with linear regression

The idea is to use the other features which don't have nulls to predict missing values.
The feature "year" is dropped as it is irrelevant and has not been discrete properly.

In this case, the prediction feature is "broad_impact", and the predictors are the rest of the features in the dataset.
We separate the dataset with no missing data rows as train set and validation set.
The Ordinary Least Squares model is used for training.



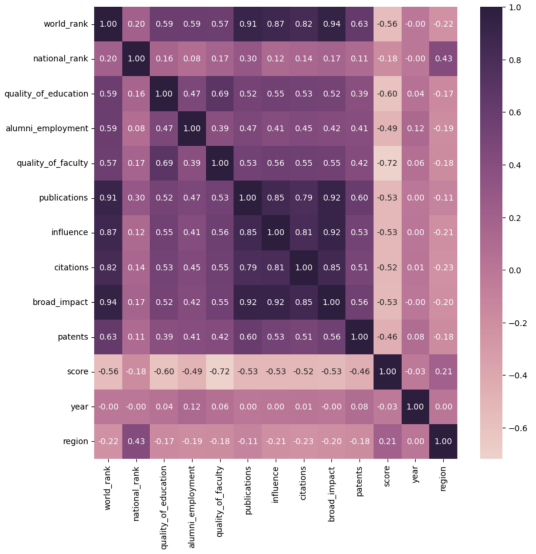

The image on the left is how the linear regression model behaves on the train and validation set.
Red: Predict values
Blue: True values

Error measure for this model is:
MS: 4330.465763325833
RMSE: 65.80627449814975
R2: 0.9480511140277353

Although the trained model does a comparatively good explanation on how "broad_impact" changed by the other independent features, the error value in prediction is huge.

The correlation matrix is visualized to help us locate why the regression model does not carry out a promising result.

We can see that high correlation exists between multiple features that should be independent.
Thus, for a quick fill-up on missing data, linear regression prediction is not the ideal solution.




### 3-2-2. Attempt to fill missing date with neighboring interpolation

Based on the result above, we can see that the "broad_impact" has a strong connection with "influence".

| influence | citations | broad_impact |
|---|---|---|
| 1 | 1 | NaN |
| 1 | 1 | NaN |
| 1 | 1 | 1.0 |
| 1 | 1 | 1.0 |
| 2 | 2 | 2.0 |
| ... | ... | ... |

Although “rank” features are correlated, we leave them out as they are more in the form of a dependent variable.
So, we will sort the data frame by the influence value (image above).

Below is the interpolation code. It basically extracts the non-nan data for the “broad_impact” feature, and interpolate the nan values between two non-nans by distance.

```
# get indices and values for non-nans
values = [(i, val) for i, val in enumerate(broad_impact_array) if not np.isnan(val)]
indices, values = zip(*values)
# use np.interp to fill in missing data
result = np.interp(np.arange(len(broad_impact_array)), indices, values)
```

### 3-2-3. Shuffling Data

Shuffling data serves the purpose of reducing variance and making sure that models remain general and overfit less[4].
Regardless if the dataset is the original one, or the one after section 3-2-2, it is following a certain order, e.g. rank and influence. In this case, shuffling data is used to make the data set fed into the train model more random.

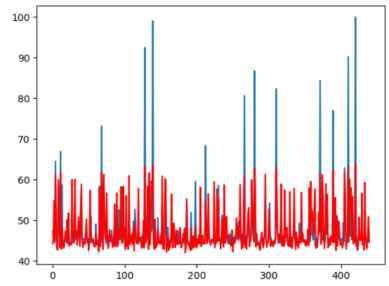
## 4 Split Test set and Trained Set

Feature “year” is dropped; it is irrelevant and has not been discrete properly.
We also remove “world_rank” and “national_rank” as they are dependent on “score” (prediction feature).
Train set and test set split up in 8:2.

## 5 Test Sklearn Linear Models

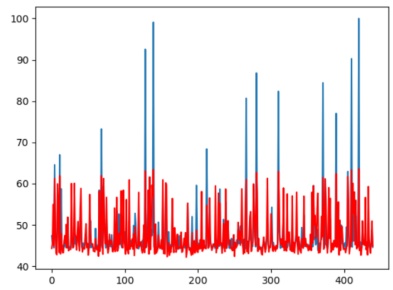
**Ordinary Least Squares**

```
coefficent is: [-0.00398048 -0.00526001 -0.06323284 -0.00063394 -0.00206063  0.02382037]
intersection is 62.460810942060604
MSE is: 27.516855639340385
RMSE is: 5.245651116814803
R2 is: 0.44577380131074074
```



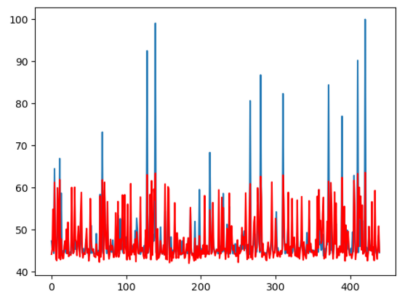

**Elastic Net CV**

```
coefficent is: [-0.0040726  -0.00528562 -0.06277226 -0.00070109 -0.00208432  0.02067641]
intersection is 62.558277567277884
MSE is: 27.49290354733072
RMSE is: 5.2433675769805 35
R2 is: 0.4462562284121204
```



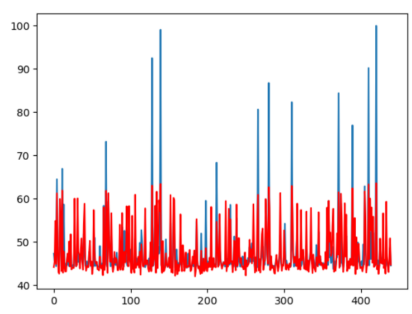

**Ridge Regression**

```
coefficent is: [-0.00398048 -0.00526001 -0.06323283 -0.00063394 -0.00206063  0.02382035]
intersection is 62.46081107031574
MSE is: 27.516855301680046
RMSE is: 5.245651084630015
R2 is: 0.44577380811167067
```




**Lasso**

```
coefficent is: [-0.00398962 -0.00526288 -0.06319346 -0.00063937 -0.00206242  0.02353893]
intersection is 62.46992474925101
MSE is: 27.51457693989505
RMSE is: 5.245433913404596
R2 is: 0.44581969735889737
```

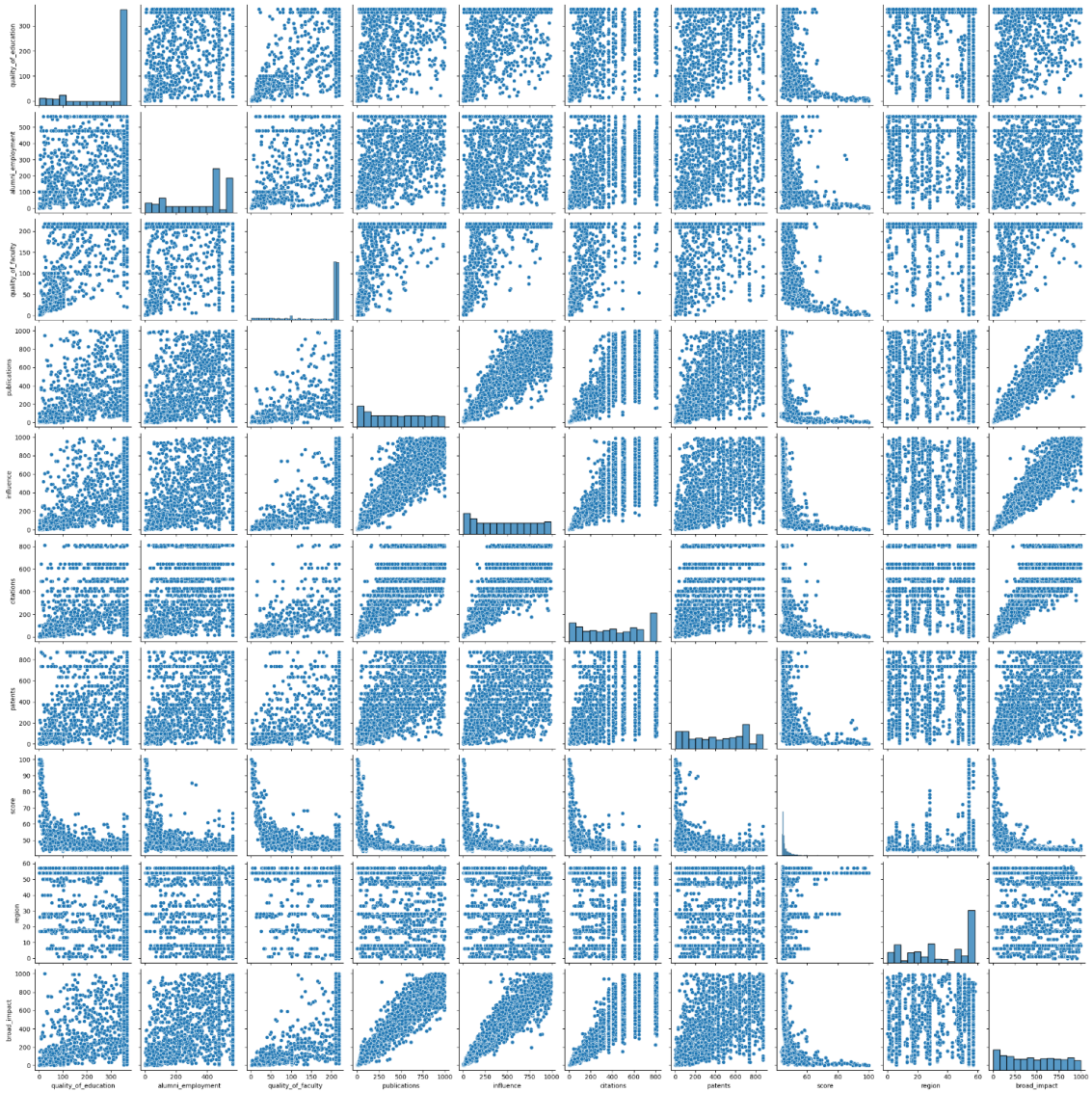


Here are the graphs to visualize the true score(blue) and prediction score(red) with the same train and test sets.

By comparing the graphs and R2, RMSE, MSE, it came to notice that different linear models do not carry out results with obvious differences. And for all four models, large errors occur when predicting scores that are comparatively high.

As mentioned in 3-2-1, high correlation exists among "independent" variables.
Thus, we need to modify the data model under collinearity situations.

## 6 Deal with Multicollinearity



Intercorrelation or multi-collinearity is the existence of predictor variables that are (highly) correlated among themselves. For example publications, influence, citations are correlated among themselves. More publications corresponds to more citations and influence, and vice-versa.

This correlation is a problem because independent variables should be independent[5]. The goal of regression is to isolate the relationship between each independent variable and the dependent variable. If an independent variable $X_1$ is highly-correlated to another independent variable $X_2$, one could not isolate the impact on the dependent variable $Y$ from $X_1$ and $X_2$.

Multicollinearity may result in:

1. coefficients become very sensitive to small changes in the model;
2. reduces the precision of the estimated coefficients.

### 6-1. Variance Inflation Factor (VIF)

VIF is a measuring tool to test multicollinearity. It measures how much the behavior (variance) of an independent variable is influenced, or inflated, by its interaction/correlation with the other independent variables.

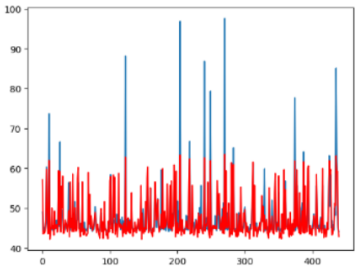
Formula is: $VIF_i = \frac{1}{1-R^2}$ , where R is the R-Squared value. Each model produces an R-squared value indicating the percentage of the variance in the individual IV that the set of IVs explains[6].

Image on the right is the VIFs for all independent variables. The region variable has low VIF values, so it can be helpful in regression. On the contrary, broad impact is highly correlated with other independent variables. Keeping it inside the dataset for training may shadow the true rule of a feature.

| | variables | VIF |
|---|---|---|
| 8 | broad_impact | 50.355761 |
| 3 | publications | 28.249336 |
| 4 | influence | 26.491687 |
| 2 | quality_of_faculty | 21.943460 |
| 0 | quality_of_education | 18.782893 |
| 5 | citations | 16.719382 |
| 1 | alumni_employment | 8.403664 |
| 6 | patents | 6.951912 |
| 7 | region | 2.741915 |

| | Original Data | Drop High VIF |
|---|---|---|
| Ordinary Least Square | MSE is: 20.417984660955785<br>RMSE is: 4.518626413076853<br>R2 is: 0.5214277043696043 | MSE is: 20.67125395578084<br>RMSE is: 4.546565072203503<br>R2 is: 0.5154913854894702 |
| Elastic Net CV | MSE is: 20.355966660524988<br>RMSE is: 4.511758710361735<br>R2 is: 0.5228813295598185 | MSE is: 20.621587810104575<br>RMSE is: 4.5410998458638385<br>R2 is: 0.5166554984881864 |
| Ridge | MSE is: 20.671253350518253<br>RMSE is: 4.54656500564088<br>R2 is: 0.5154913996760764 | MSE is: 20.671253350518253<br>RMSE is: 4.54656500564088<br>R2 is: 0.5154913996760764 |
| Lasso | MSE is: 20.666769345206053<br>RMSE is: 4.546071858781607<br>R2 is: 0.5155964992121722 | MSE is: 20.666769345206053<br>RMSE is: 4.546071858781607<br>R2 is: 0.5155964992121722 |

Red: Prediction
Blue: True




Plots for true values and predict values comparison.
All 8 runs on the same test set and train set in the chart follows this distribution.

Here, we compare and contrast the linear regression results trained by the original dataset and by the dataset without high-VIF features. We can see from the table that dropping independent variables does not bring in obvious improvement for any of the regression models.

Prediction results on high scores are still in low accuracy.

In the following steps, we will try:
1. Other ways to deal with multicollinearity
2. Observing data again and see if there are other factors affecting the learning

### 6-2. Lasso Select Features [8]

In statistics and machine learning, lasso (least absolute shrinkage and selection operator; also Lasso or LASSO) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model[7].

```
# use Lasso to extract features
feature_data = data_frame.drop(columns=["score"])
feature_names = feature_data.columns
y = Y

lasso_model = linear_model.Lasso(alpha = 10)
select_model = SelectFromModel(lasso_model)

select_model.fit(feature_data,y)
select_features = feature_names[select_model.get_support()]
```

However, there are still no improvements on regression performances after selecting features by the Lasso model. For this reason, we will consider feature expansion to test if the true rule of this model is indeed $Y = X\beta + \epsilon$.

### 6-3. Feature Expansion

Since we have multicollinearity among the independent variables, for linear regression to work, we can assume that the actual rule between dependent and independent variables is “equivalent to a linear classification problem in some higher dimensional feature space”[9].

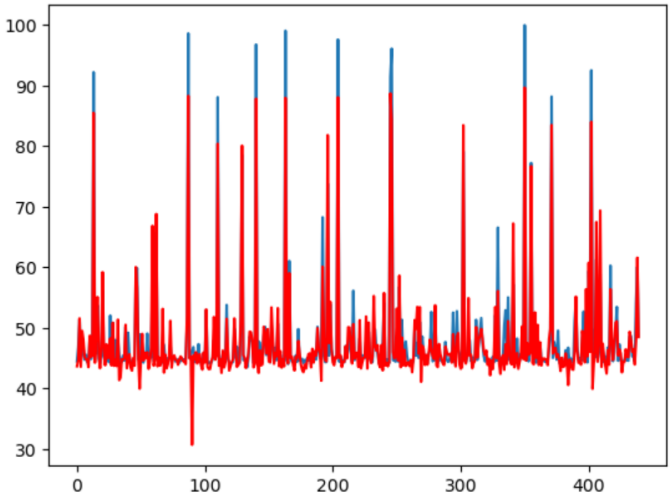
*“sklearn.preprocessing.PolynomialFeatures”* is used in this trial.

| | degree=2 | degree=3 | degree=4 | With Original Data: |
|---|---|---|---|---|
| Ordinary Least Square | MSE is: 11.222190412904084<br>RMSE is: 3.349953792652084<br>R2 is: 0.7742478980389744 | MSE is: 5.165205888154584<br>RMSE is: 2.272708931683638<br>R2 is: 0.8960937175890771 | MSE is: 4.9237523836330945<br>RMSE is: 2.2189529926596223<br>R2 is: 0.9009509365602419 | MSE is: 20.417984660955785<br>RMSE is: 4.518626413076853<br>R2 is: 0.5214277043696043 |

The error estimations are compared between running OLS on the original dataset, and OLS on the dataset with different expansion degrees.

Once the feature dimension got expanded, we can see that:
1. Error estimation decreases
2. Model explains the dependent feature better.

The performance of regression keeps improving while the expansion degree gets higher. Below is a graph comparing predicted values and true values with expansion degree at 3.




Red: predicted values;
Blue: true values.

Before feature expansion, once the score value is above 60, the prediction results have huge errors between true values.

Here, we can see that although it is still difficult to predict higher scores, the performance quality only gets lower for scores over 80. However, the downside is that the prediction on lower scores also starts to get unstable.

**6-4. Outlying Observations**

Since in the previous trials, the model always behaves in a low accuracy on high score predictions, we suspect that there exists outlying observations in samples.

The definitions of outliers presented are subjective. More objectively, for the one-dimensional case, one of the options adopted in the sciences in general is to take as outliers the observations that deviate more than a certain amount of their standard deviation σ around its respective mean μ[10]。

Not all outliers should be or can be removed. They can be informative and represent the true nature of the model rules. Here we will test how the regression can be affected by these outliers.

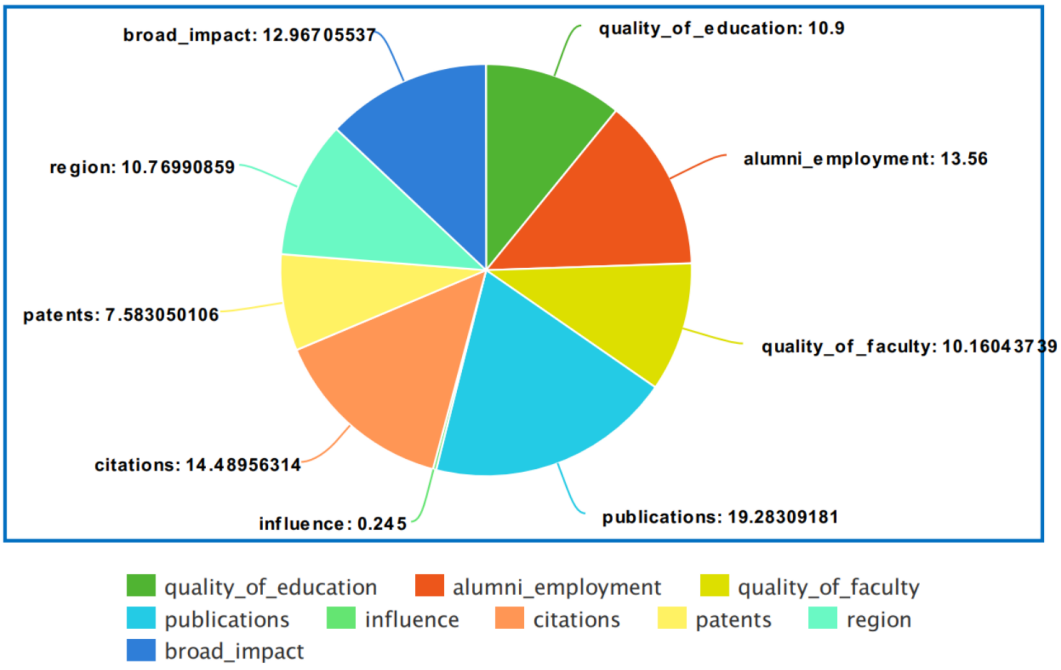
A common way of testing outlying observations is $3\sigma - rule$.

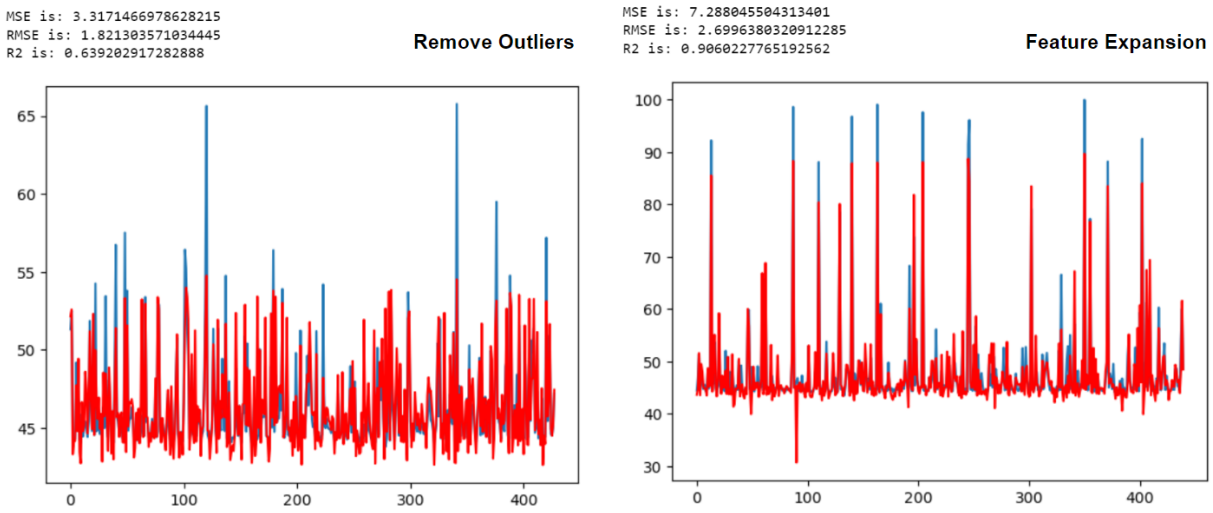
```
def three_sigma(feature):
    """return indices for elements that not fall in the 3σ range"""
    rule = (np.mean(feature)-3*feature.std()>feature) | (np.mean(feature)+3*feature.std()<feature)
    index = np.arange(feature.shape[0])[rule]
    return index

def out_three_sigma(data):
    to_del_indices = set()
    new_data = data.copy()
    for col in data.columns:
        del_indices = three_sigma(data[col])
        to_del_indices.update(del_indices)
    new_data = new_data.drop(new_data.index[i] for i in to_del_indices)
    return new_data
```

We removed the samples that not fall in the $3\sigma$ range.
There are other ways to deal with outliers. For example, replace the outlying elements with $Nan$. Then fill in the $Nan$ by regression(Section 3-2-1) or mean or interpolation(Section 3-2-2) etc..

The regression result here can give us a better understanding on how independent variables affect the scores below 70.




MSE and RMSE for outlier removal is lower than the feature expansion. R2, however, is on the contrary. The reason behind this is that the regression model behaves badly at high scores, but by removing outliers these high values are no longer taken into consideration. The R2 gets lower because the removal results in loss of data. The model gives good regression for scores that are not too high or too low. But it doesn't explain the scores out of this range.

Compared to run models on the original dataset, it is an interesting fact that by dropping the outliers, without any extra process on data, the regression result got improved massively.

Thus, we will choose expansion for data processing, or perhaps the combination of expansion and outlier removal.

## 7 Final Model Build and Evaluation

### 7-1 Compared Regression at Different Expansion Degree

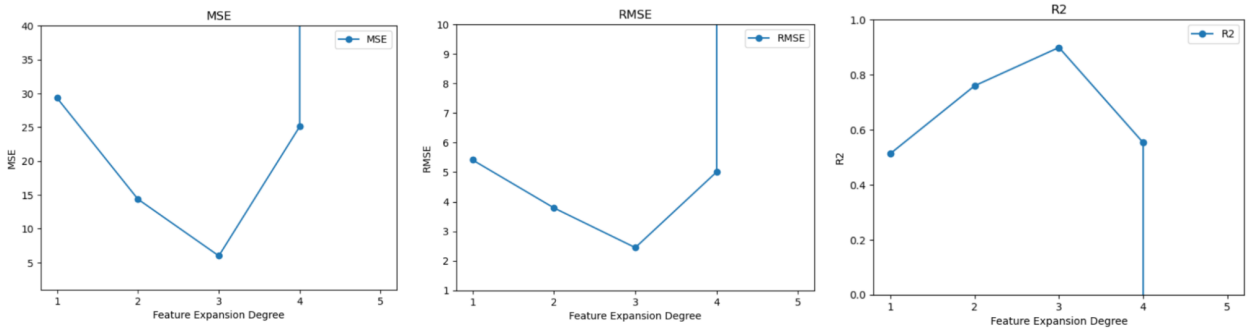
In this section, we will analyze results on different expansion degrees with cross validation.

```
model = linear_model.LinearRegression()

mean_r2 = []
mean_mse = []
mean_rmse = []

for degree_v in [1, 2, 3, 4, 5]:
    # feature expansion
    feature_data = data_frame.drop(columns=["score"])
    polynomy = PolynomialFeatures(degree=degree_v)
    new_feature_data = polynomy.fit_transform(feature_data)
    Y = data_frame["score"]
    # cross validation
    r2_scores = cross_val_score(model, new_feature_data, Y, cv=5, scoring ='r2')
    mse_scores = np.abs(cross_val_score(model, new_feature_data, Y, cv=5, scoring ='neg_mean_squared_error'))
    mse = np.mean(mse_scores)

    mean_r2.append(np.mean(r2_scores))
    mean_mse.append(mse)
    mean_rmse.append(np.power(mse, 0.5))
```




Mean squared error (MSE) measures the amount of error in statistical models.
R-squared(R2) measures how well the trained model fits the data.

In short, the greater the MSE and RMSE is, the weaker the model is to predict dependent variables.
The greater the R2 is, the better the model explains the changes in the dependent variable by independent variables.

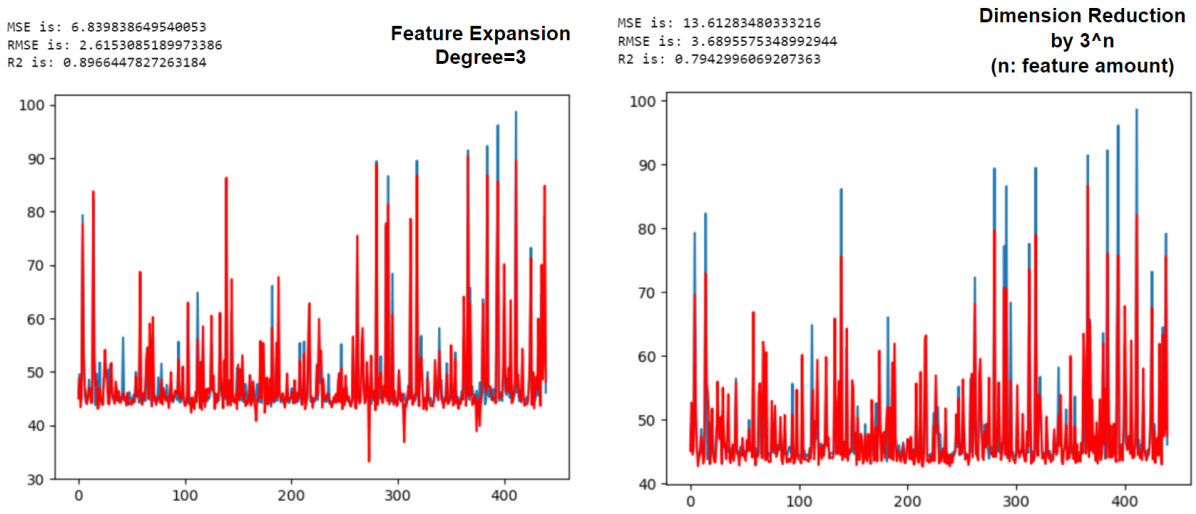
When the feature expansion degree grows from 1 -3, the trained model gradually gets better on performance.
After the degree gets greater than 3, the model prediction has a sharp decline, especially when the degree gets to 5.

This suggests that the independent features and dependent features do not entirely follows a linear regression model: $Y = X\beta + \epsilon$

But more like: $Y = X^3\beta + \epsilon$ *or* $Y = X^4\beta + \varepsilon (X\ is\ the\ demision\ reduction\ of\ original\ X)$

### 7-2 Dimension Reduction vs Feature Expansion

Based on the conclusion above, in this section, we compare the regression performance between dimension deduction(np.cbrt()) and feature expansion.




In general, feature expansion is the go-to-solution for a better linear regression.
Feature reduction gives promising results, but still lack of accuracy to predict high scores.

## 8 Summary

Main features that affect the dependent variable “score” are: publications, citations, alumni employment. Region can have a comparatively large impact on “score”. This is possibly because region values are discrete and do not have much correlation with other independent variables.

Through this practice, we understand that a good dataset is a foundation for linear regression results. Ways of optimizing data, such as: VIF, outliers, feature expansion, dimension reduction et.c are tested. Among which feature expansion carries out the best regression result with low RMSE and high R-Squared (especially for explaining high scores). It is also important to choose data optimization methods based on the nature of the given dataset.

## 9 References

[1] https://en.wikipedia.org/wiki/Linear_regression

[2] Convert categorical string data into numeric:
https://www.geeksforgeeks.org/how-to-convert-categorical-string-data-into-numeric-in-python/

[3] 7 Ways to Handle Missing Values in Machine Learning:
https://towardsdatascience.com/7-ways-to-handle-missing-values-in-machine-learning-1a6326adf79e

[4] Why should the data be shuffled for machine learning task:
https://datascience.stackexchange.com/questions/24511/why-should-the-data-be-shuffled-for-machine-learning-tasks

[5] https://statisticsbyjim.com/regression/multicollinearity-in-regression-analysis/

[6] https://statisticsbyjim.com/regression/variance-inflation-factors/

[7] https://en.wikipedia.org/wiki/Lasso_(statistics)

[8] A Critical Review of LASSO and Its Derivatives for Variable Selection Under Dependence Among Covariates: https://onlinelibrary.wiley.com/doi/full/10.1111/insr.12469

[9] Yao, K. et al. (2003) 'Feature expansion and feature selection for General Pattern Recognition problems', International Conference on Neural Networks and Signal Processing, 2003. Proceedings of the 2003 [Preprint]. doi:10.1109/icnnsp.2003.1279205.

[10] https://www.scielo.br/j/bcg/a/kPytvJQWxC5ZNjm987zmnVy/?lang=en