Preprocessing
Introduction
Feature Engineering
Averaging Collinear Distributions
Non-parametric Approach: kNN
Parametric Approach: Simple Linear Regression

A Comparison of Parametric and Nonparametric Approaches

using R and python.

Yusuf Baran Tanrıverdi[1]

University of Cassino and Southern Lazio[1],

May, 2023

MAIA

Preprocessing
Introduction
Feature Engineering
Averaging Collinear Distributions
Non-parametric Approach: kNN
Parametric Approach: Simple Linear Regression

Introduction

- **Objective**: Evaluate the performance of kNN and LM models on a regression task
- **Key points**: Feature correlation monitoring is crucial to build the right model. That means, scatter matrix is a life saver!

Challenges

- Small dataset! We do not know if the model is validated.
- Features are so homogeneously distributed. Almost constant!
- Regression over one feature ("v3") is seemingly over powerful, this could cause to overfitting for real data.

Scatter Matrix using seaborn from Python

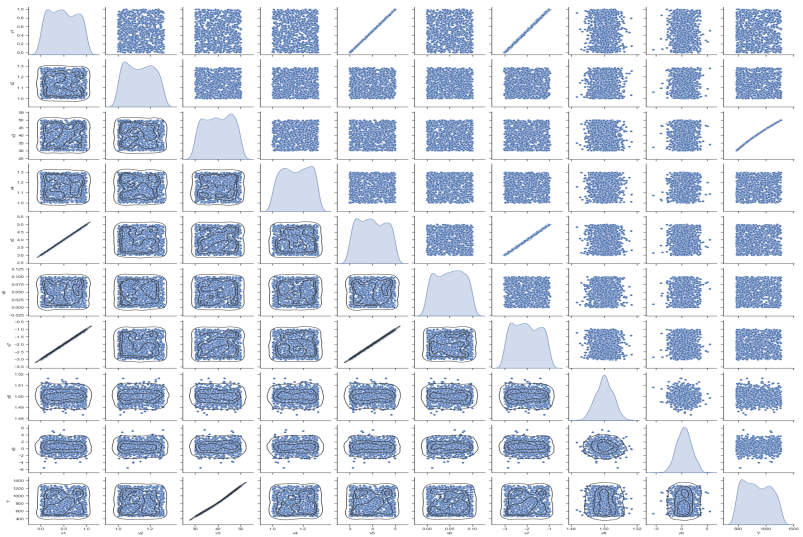


Figure: Scatter matrix including all vectors

Preprocessing
Introduction
Feature Engineering
Averaging Collinear Distributions
Non-parametric Approach: kNN
Parametric Approach: Simple Linear Regression

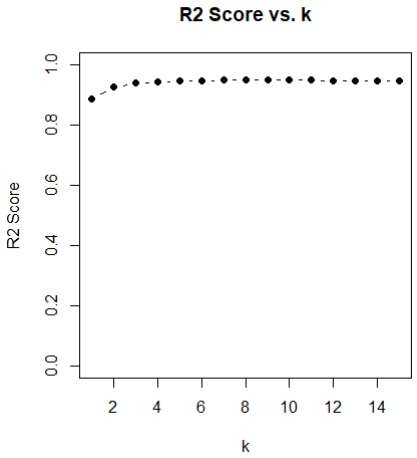
Feature Combination

- Since *v1*, *v5* and *v7* are hugely correlated, I averaged them and create a combined feature. Then, they are dropped from the dataset - to avoid repetition.
- Also, a distribution similarity between *v8* and *v9* is also observed. They are summed too.
- Lastly, *v2* and *v6* are not included in the regression because they do not affect the **Y** variable at all.
- I also normalized in a standard way.

```
# Scatter-matrix inspired and empirically tested selected features.
regress_cols <- c("v3", "avg_v1_v5_v7", "avg_v8_v9")
```

Tuning the kNN: How to choose k variable

Considering the R2 score as the fitness of the kNN regression cross-validation, I plotted the iteration. k=10 has given the best outcome.



Simple caret Linear Regressor & Conclusion

Here, I fitted the model with a simple *lm* regressor. Applying repeated cross-validation with k=10, n= 10; I have the best results. The results are climbing up to the perfect classifier status. Below, I provide the scores for both approaches:

Table: Performance Results

Model	R2Score	RMSE
caret::lm	0.996	0.63
FNN:knn.reg	0.993	0.82

- High performance obtained, with linear regressor outperforming slightly.
- The scatter matrix, generated using Python's seaborn library, provided valuable insights into the relationships between features.
- The challenges encountered included a small dataset with uncertain validation, homogeneously distributed features, and the potential for overfitting.