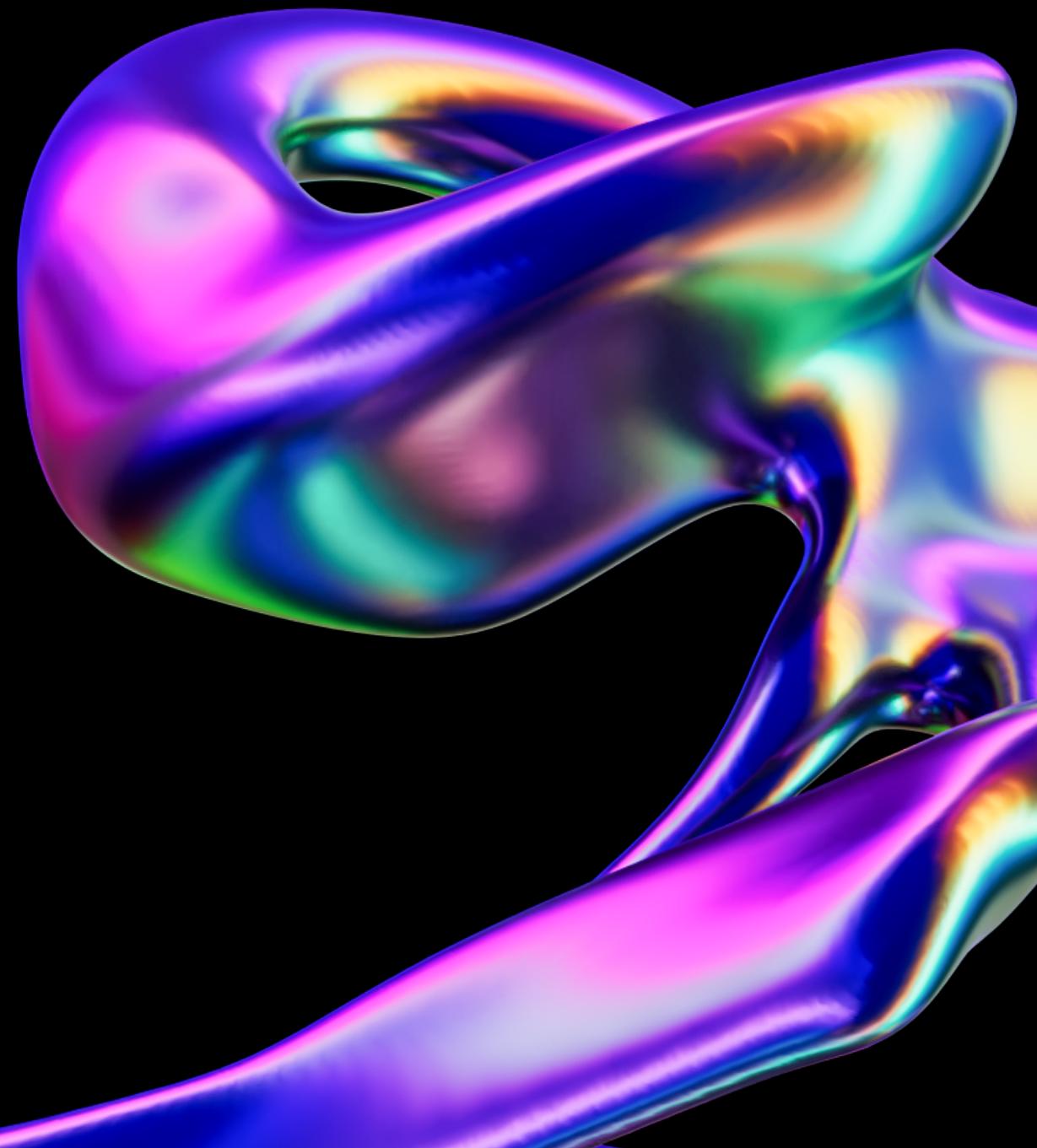


# Machine Learning Project



# Agenda

- **NUMERICAL**
  - INTRODUCTION
  - DATASET
  - DESCRIPTBITION
  - PREPROSSESING
  - ML MODELS
- **IMAGE**
  - INTRODUCTION
  - DATASET
  - DESCRIPTBITION
  - PREPROSSESING
  - ML MODELS



1

**House Sales  
in King  
County, USA**

# Introduction

- IN THIS REPORT, WE PRESENT A MACHINE LEARNING MODEL IS DEVELOPED FOR PREDICTING HOUSE PRICES USING LINEAR REGRESSION AND KNN.

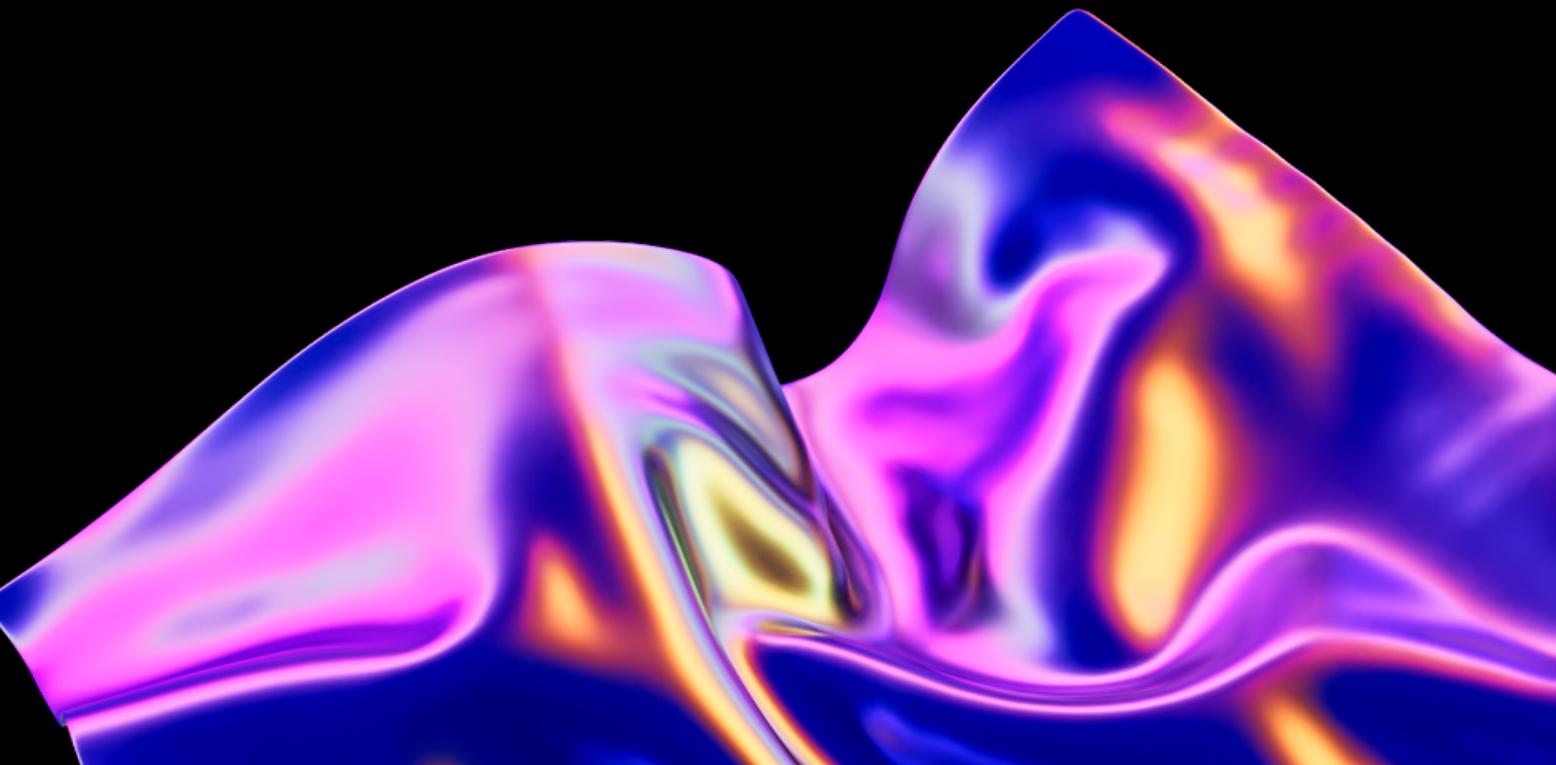
# Numerical dataSet

House Sales in King County, USA



# Data Description

- THIS DATASET CONTAINS HOUSE SALE PRICES FOR KING COUNTY, WHICH INCLUDES SEATTLE. IT INCLUDES HOMES SOLD BETWEEN MAY 2014 AND MAY 2015.



# Features Description

- Id : Unique ID for each home sold
- Date : Date of the home sale
- Price : Price of each home sold
- Bedrooms : Number of bedrooms
- Bathrooms : Number of bathrooms, where accounts for a room with a toilet but no shower

# Features Description

- **sqft\_living** : Square footage of the apartments interior living space
- **sqft\_lot** : Square footage of the land space
- **waterfront** : A dummy variable for whether the apartment was overlooking the waterfront or not

# Features Description

- View : An index from 0 to 4 of how good the view of the property was
- Condition : An index from 1 to 5 on the condition of the apartment
- Grade : An index from 1 to 13, where 1–3 falls short of building construction and design, 7 has an average level of construction and design, and 11–13 have a high quality level of construction and design

# Features Description

- Sqft\_above : The square footage of the interior housing space that is above ground level
- Sqft\_basement : The square footage of the interior housing space that is below ground level
- Yr\_built : The year the house was initially built
- Yr\_renovated : Square footage of the land space

# Features Description

- Sqft\_basement : The square footage of the interior housing space that is below ground level
- Yr\_built : The year the house was initially built
- Yr\_renovated : Square footage of the land space
- Zipcode : What zipcode area the house is in
- Lat : Latitude
- Lang : Longitude

# Data Before Preprocessing

- SIZE OF DATA : 2.52 MB
- NUMBER OF FEATURES : 21 FEATURE
- NUMBER OF SAMPLES : 21613 SAMPLE

# Data Preprocessing

```
54]: df["yr_renovated"].value_counts().reset_index()  
  
54]:

| yr_renovated | count |
|--------------|-------|
| 0            | 20699 |
| 1            | 2014  |
| 2            | 2013  |
| 3            | 2003  |
| 4            | 2005  |
| ...          | ...   |
| 65           | 1951  |
| 66           | 1959  |
| 67           | 1948  |
| 68           | 1954  |
| 69           | 1944  |


```

```
[52]: df["waterfront"].value_counts().reset_index()  
  
[52]:

| waterfront | count |
|------------|-------|
| 0          | 21450 |
| 1          | 163   |

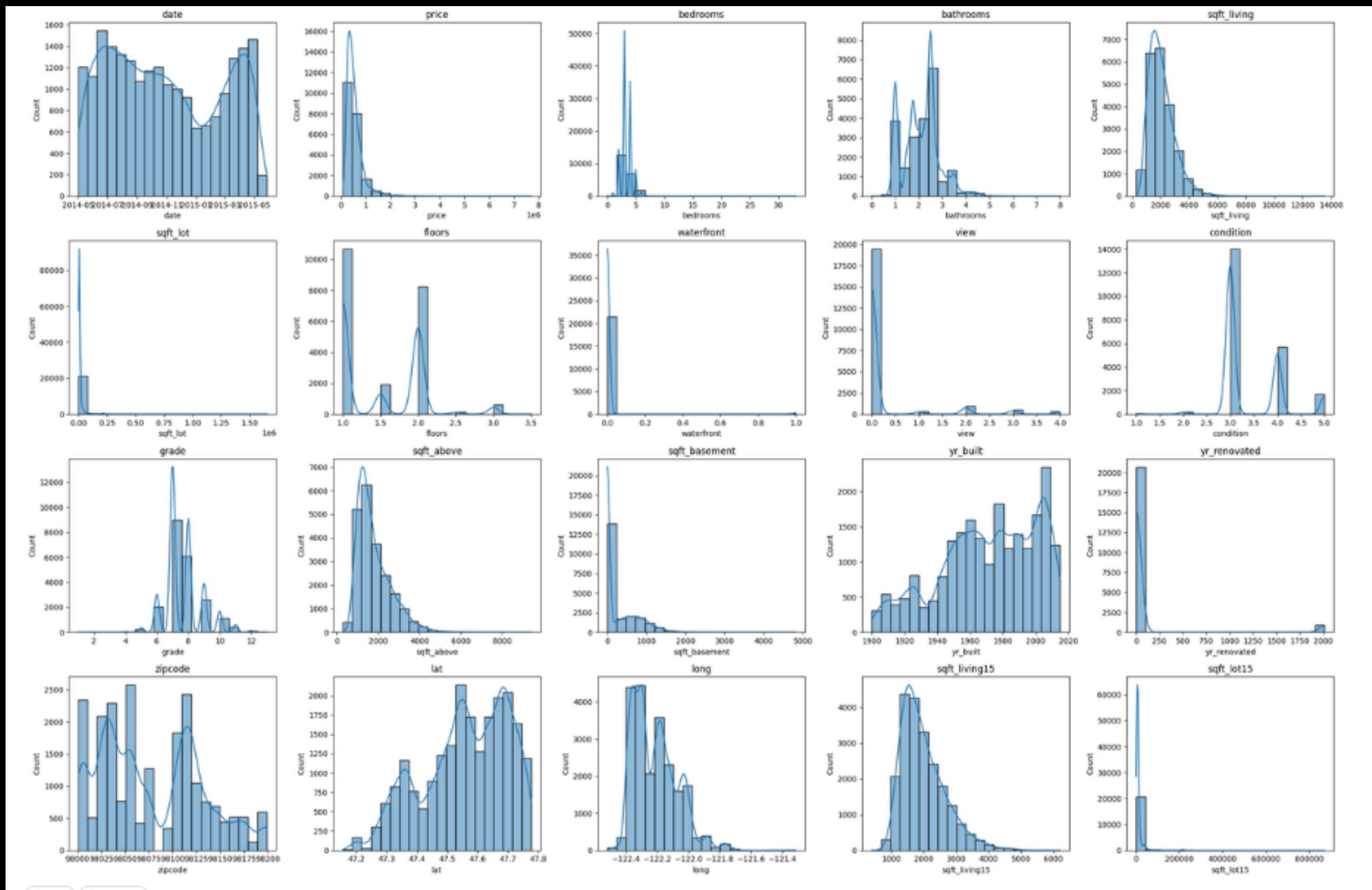
  
  
[53]: df["view"].value_counts().reset_index()  
  
[53]:

| view | count |
|------|-------|
| 0    | 19489 |
| 1    | 963   |
| 2    | 510   |
| 3    | 332   |


```

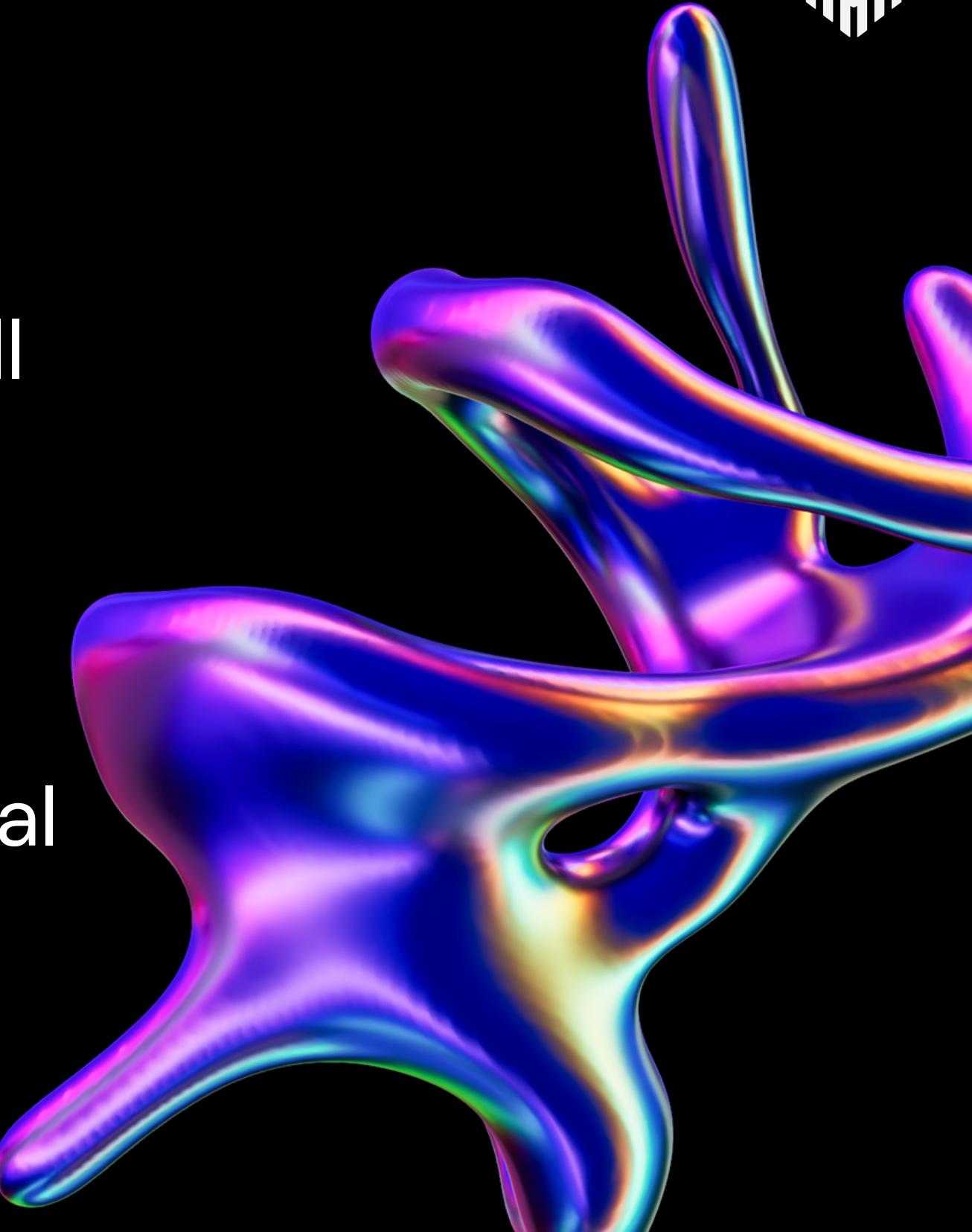
Most Values of these features (yr\_renovated, view, waterfront) are zero ,so we gonna drop them

# Data Distribution (Before)



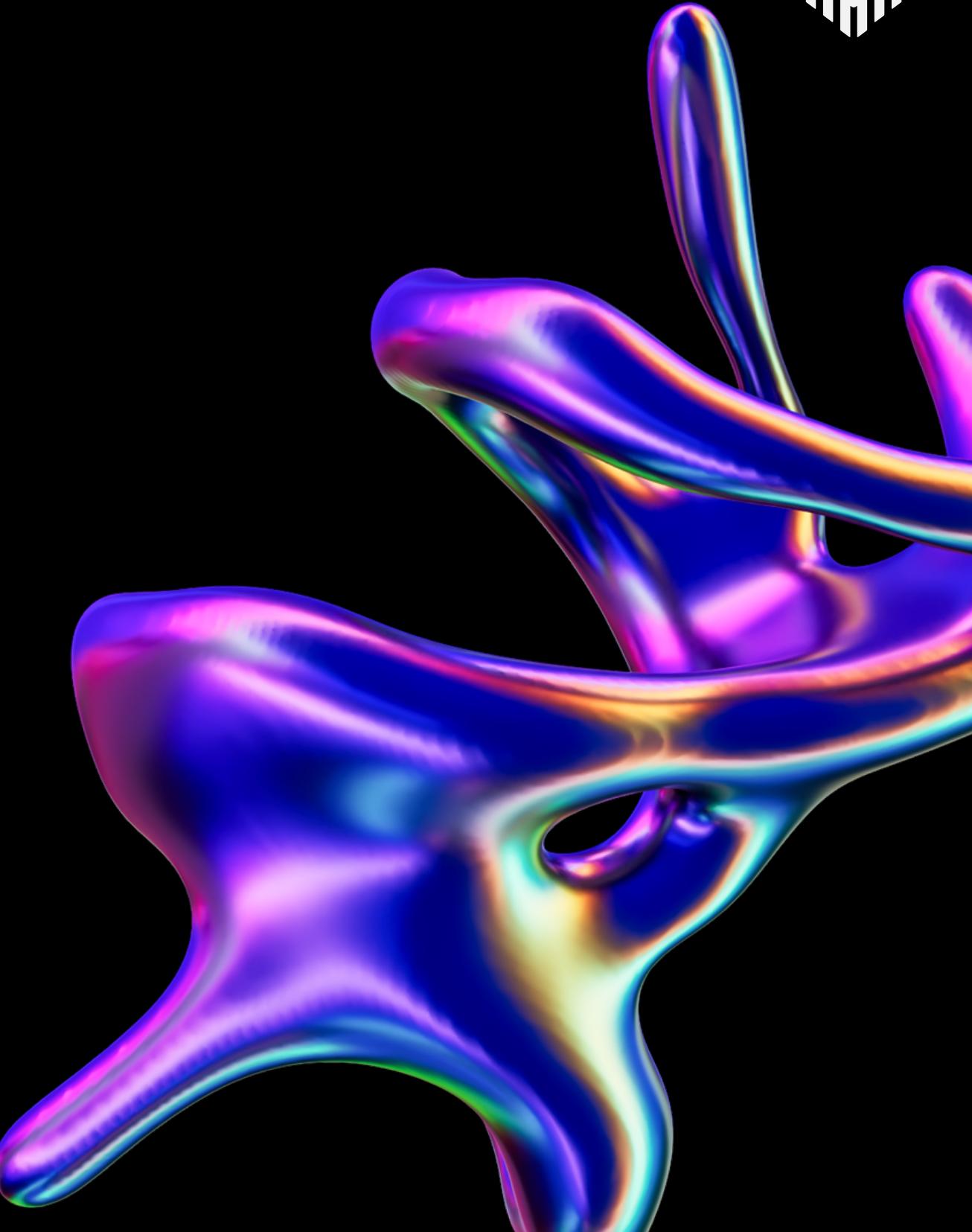
# Standard Scaler

- We scaling data with Standard Scaler To all column and ignore this column:
- "grade", "condition", "floors", "bathrooms", "bedrooms", "date"
- Because values in this column is categorical

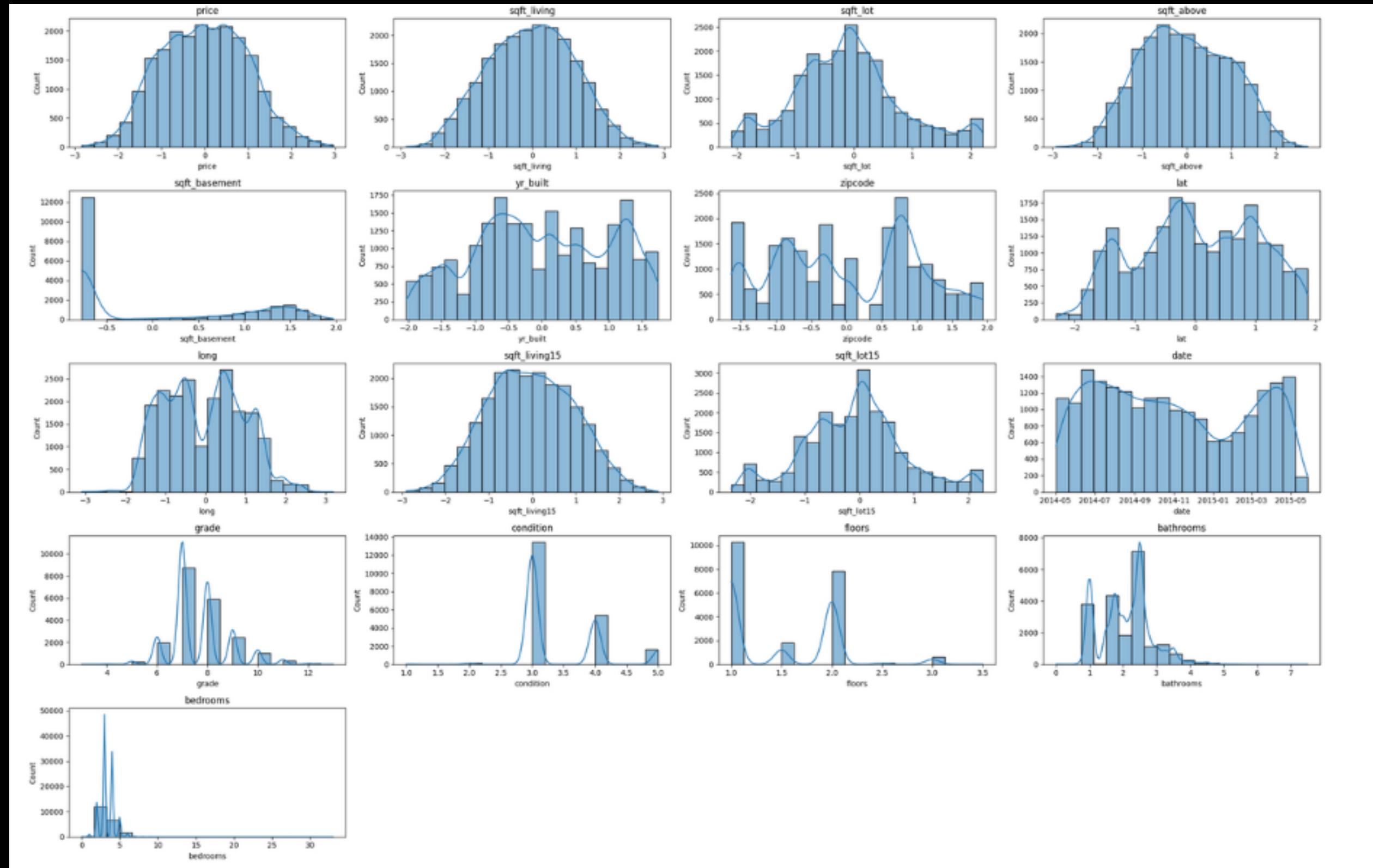


# Transform

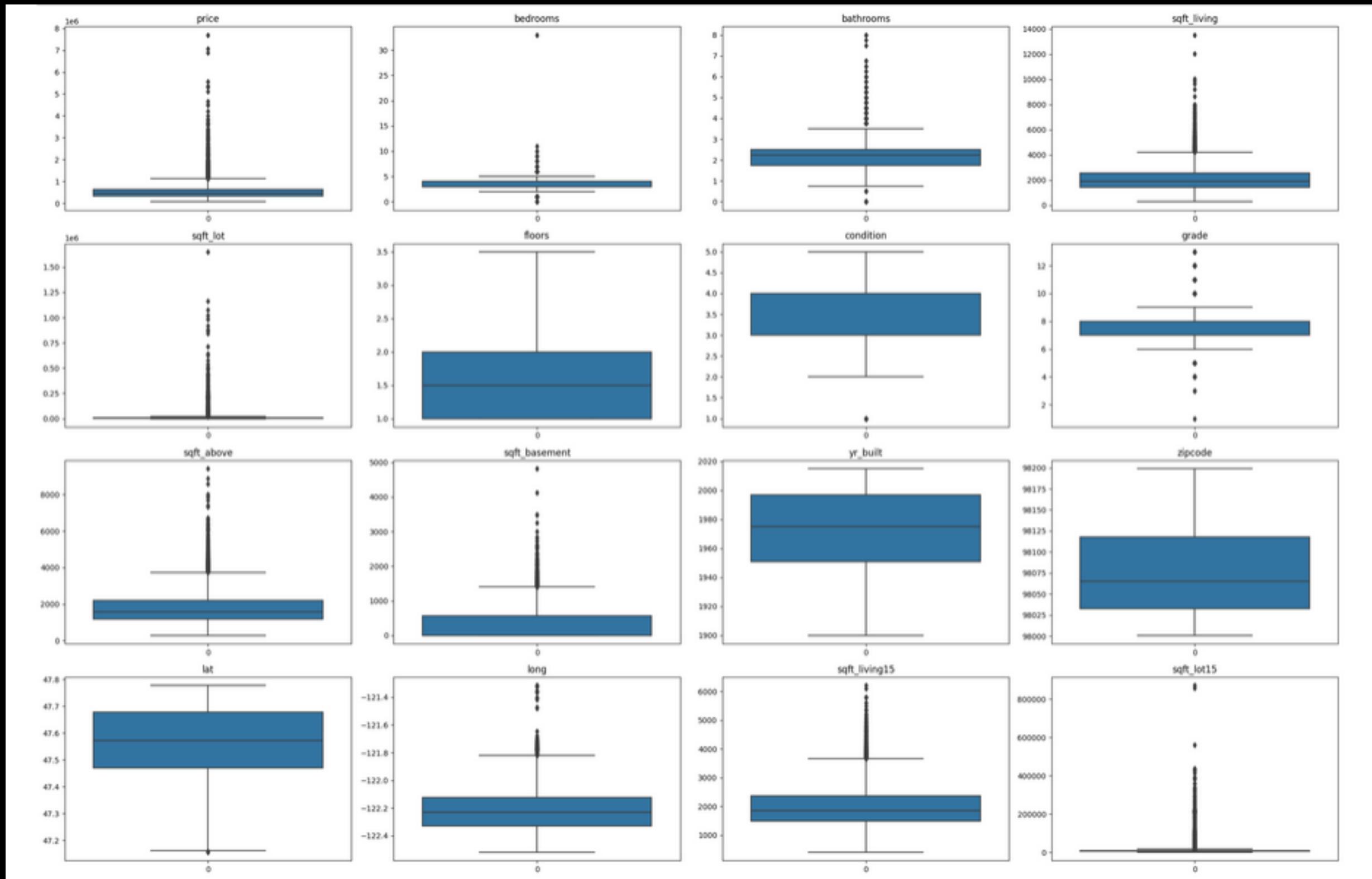
- transform data with power transform (method = yeo-johnson ) to make distribution of data normalized



# Data Distribution (after)

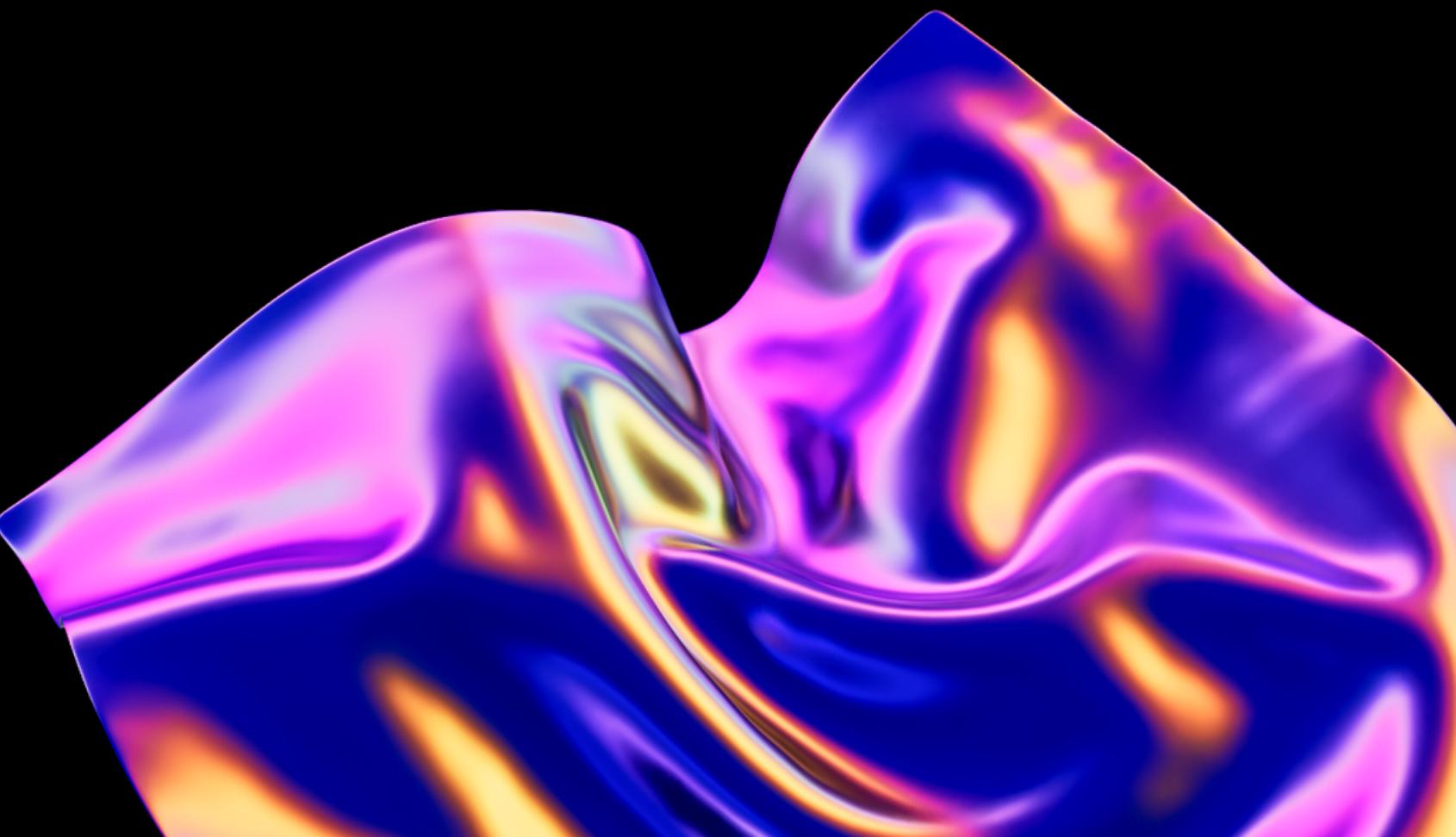


# Outliers (Before)

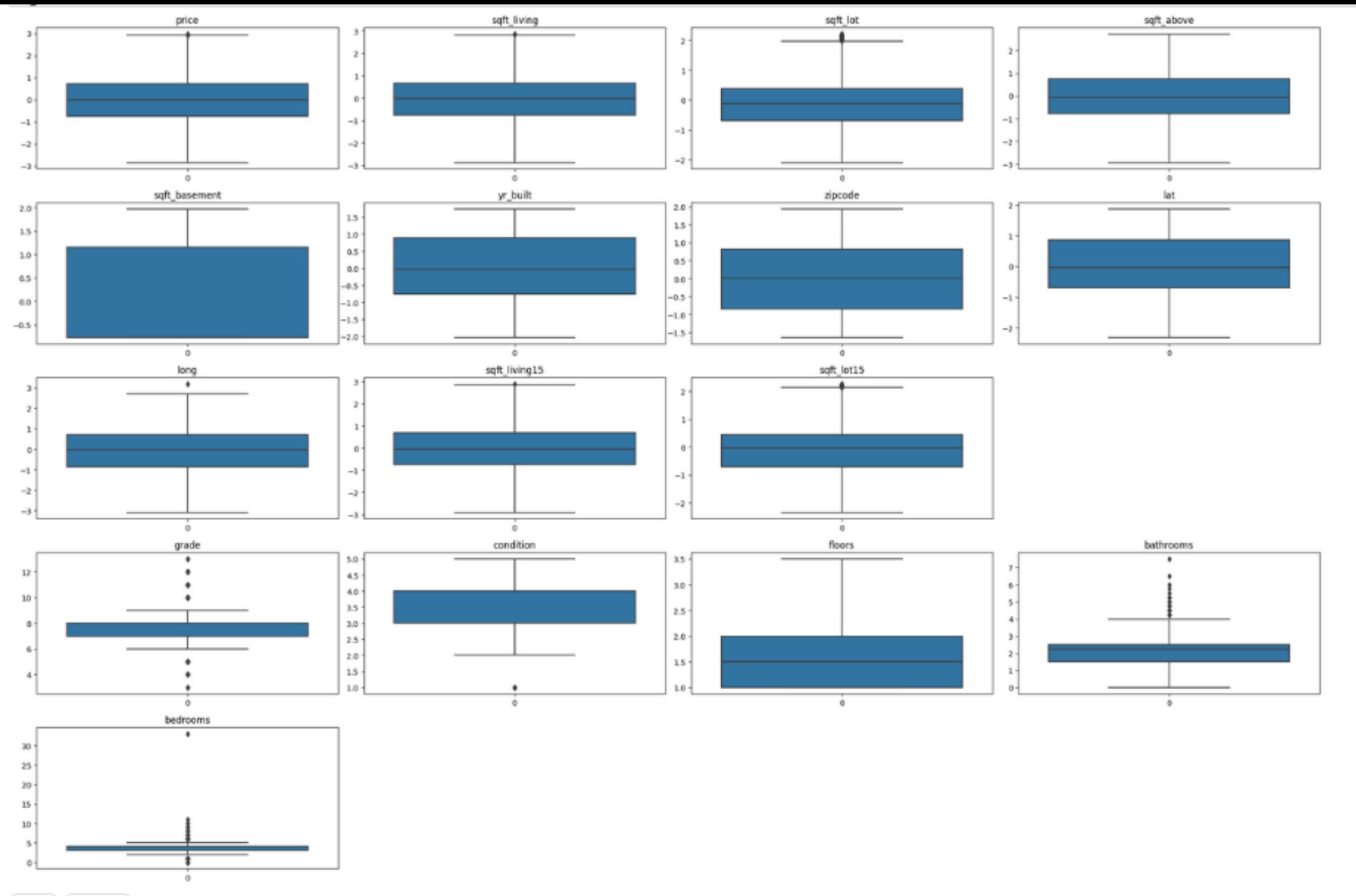


# Handle Outliers

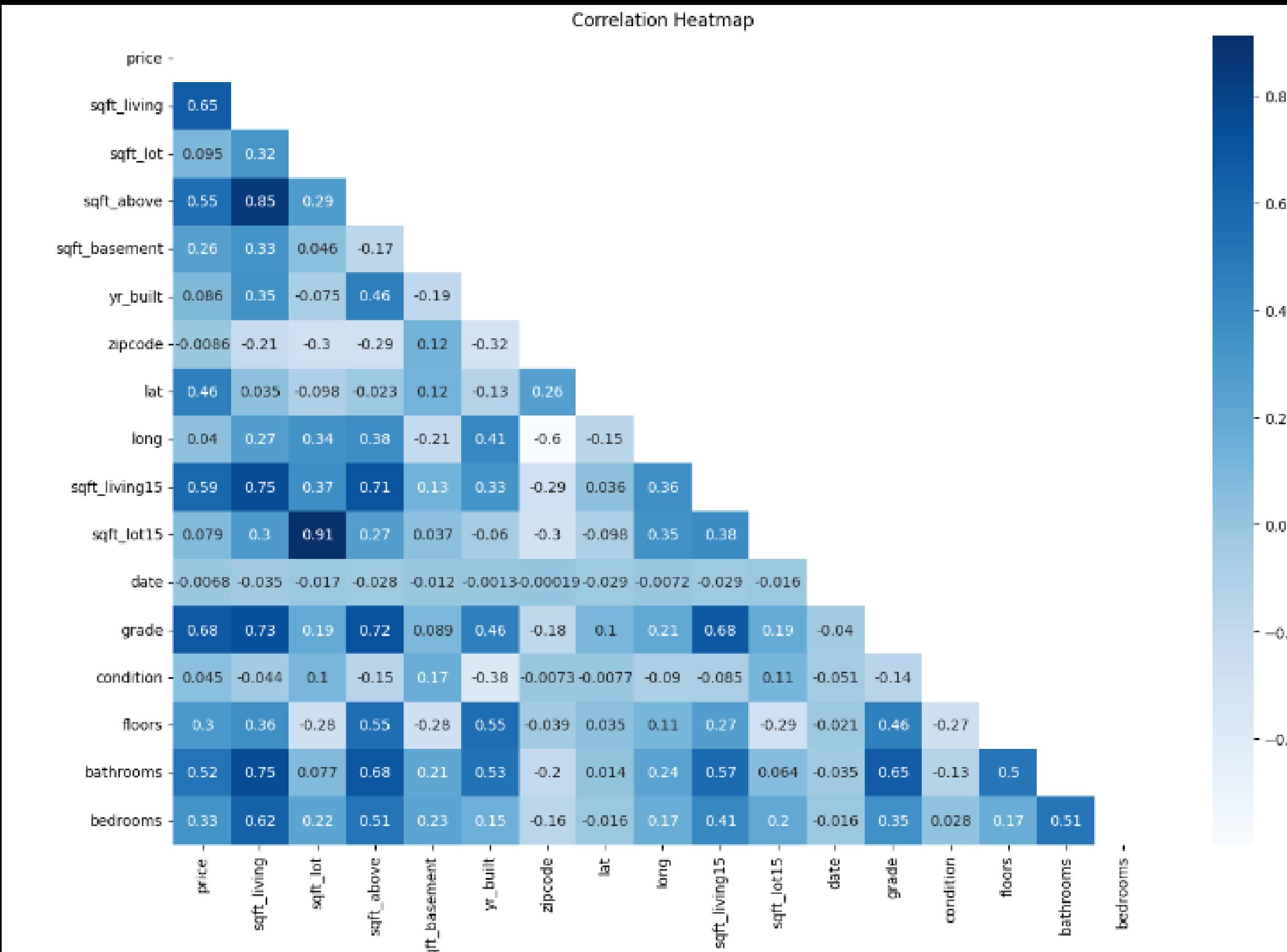
We Are Using IQR Technique To Handle  
Outliers And Remove Outliers



# Outliers (After)

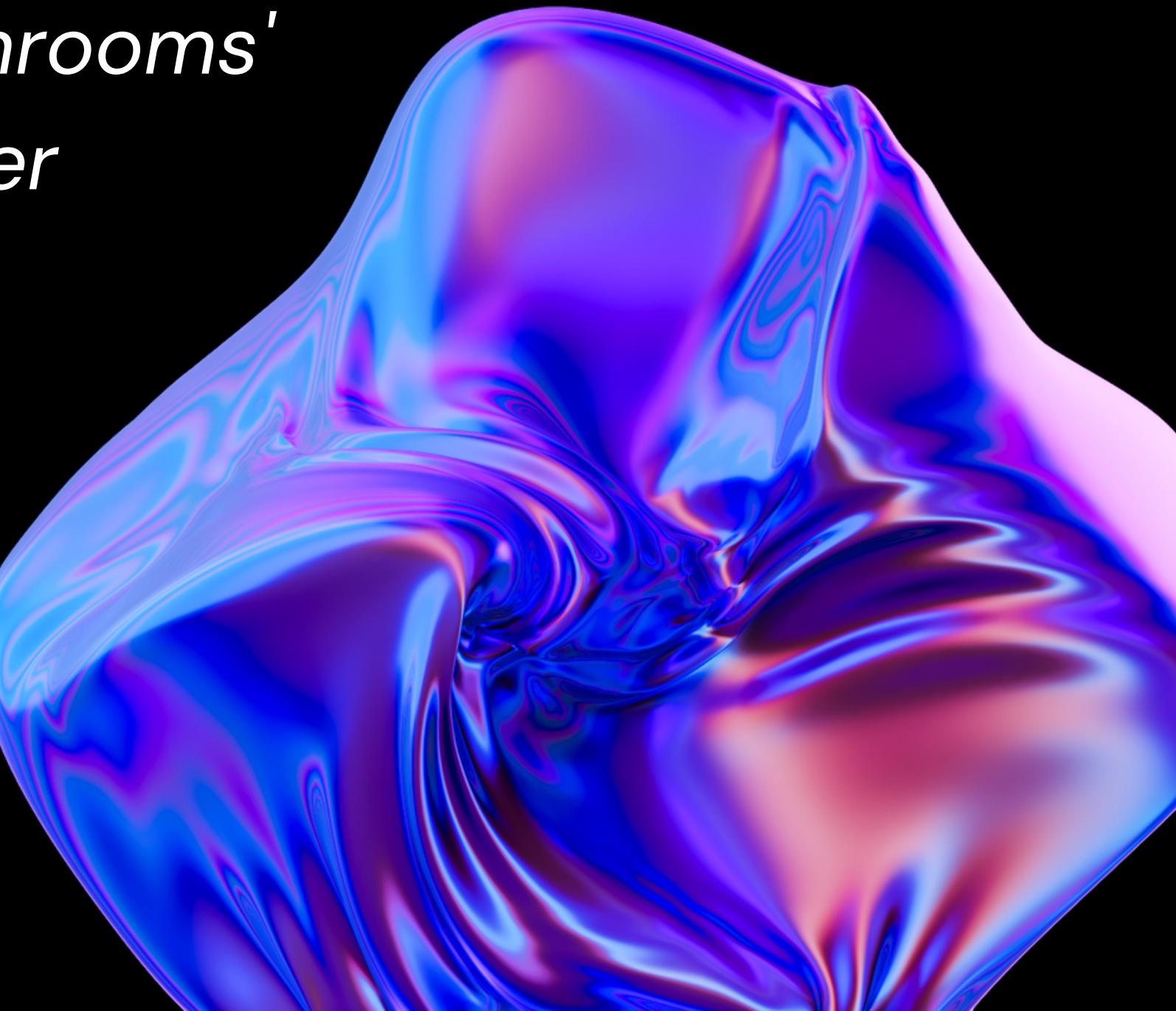


# correlation (Before)

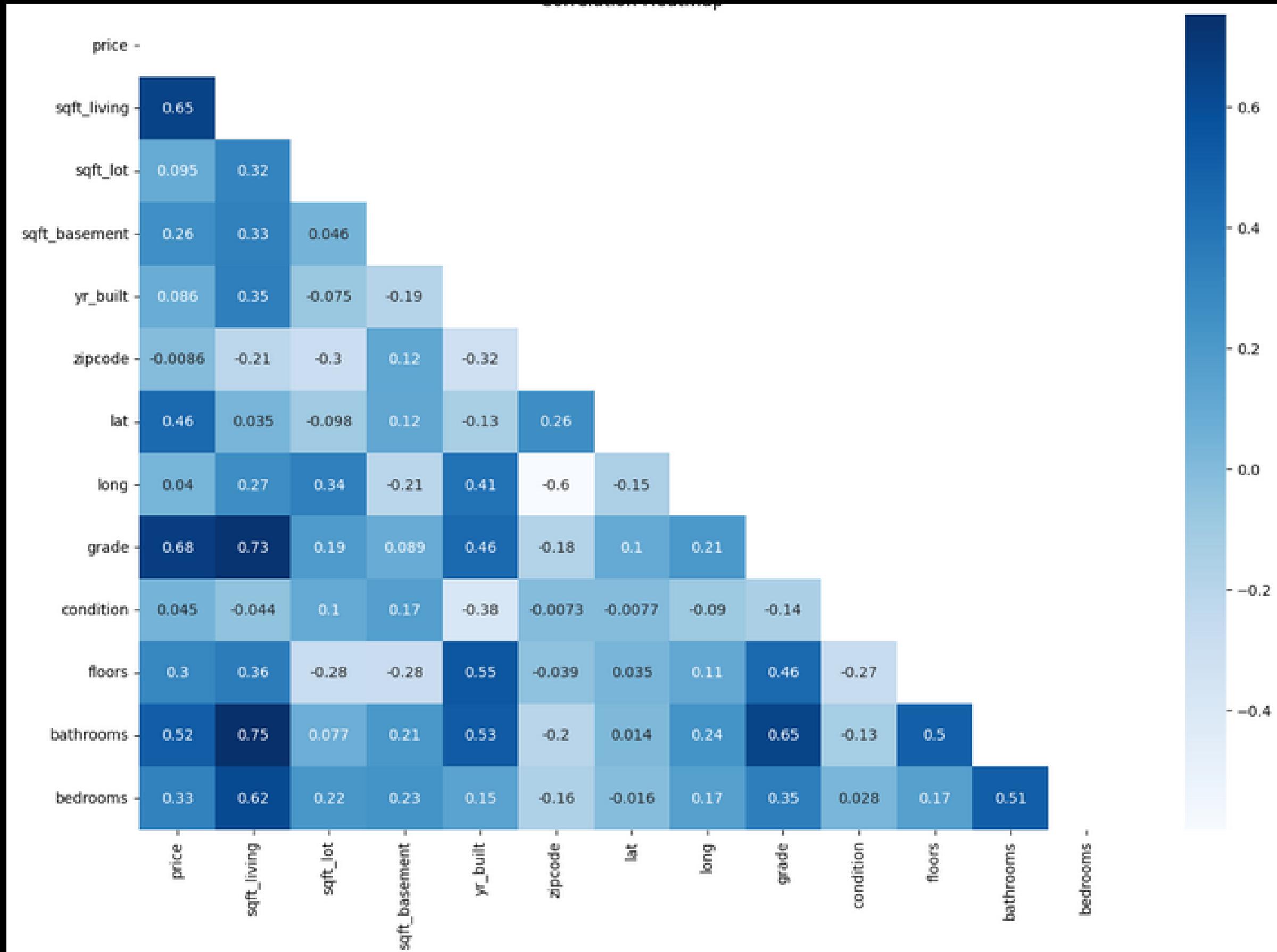


# Multicollinearity

- we droped these columns  
`'sqft_above','sqft_living15','grade','bathrooms'`
- Due to *high corellation with each other*

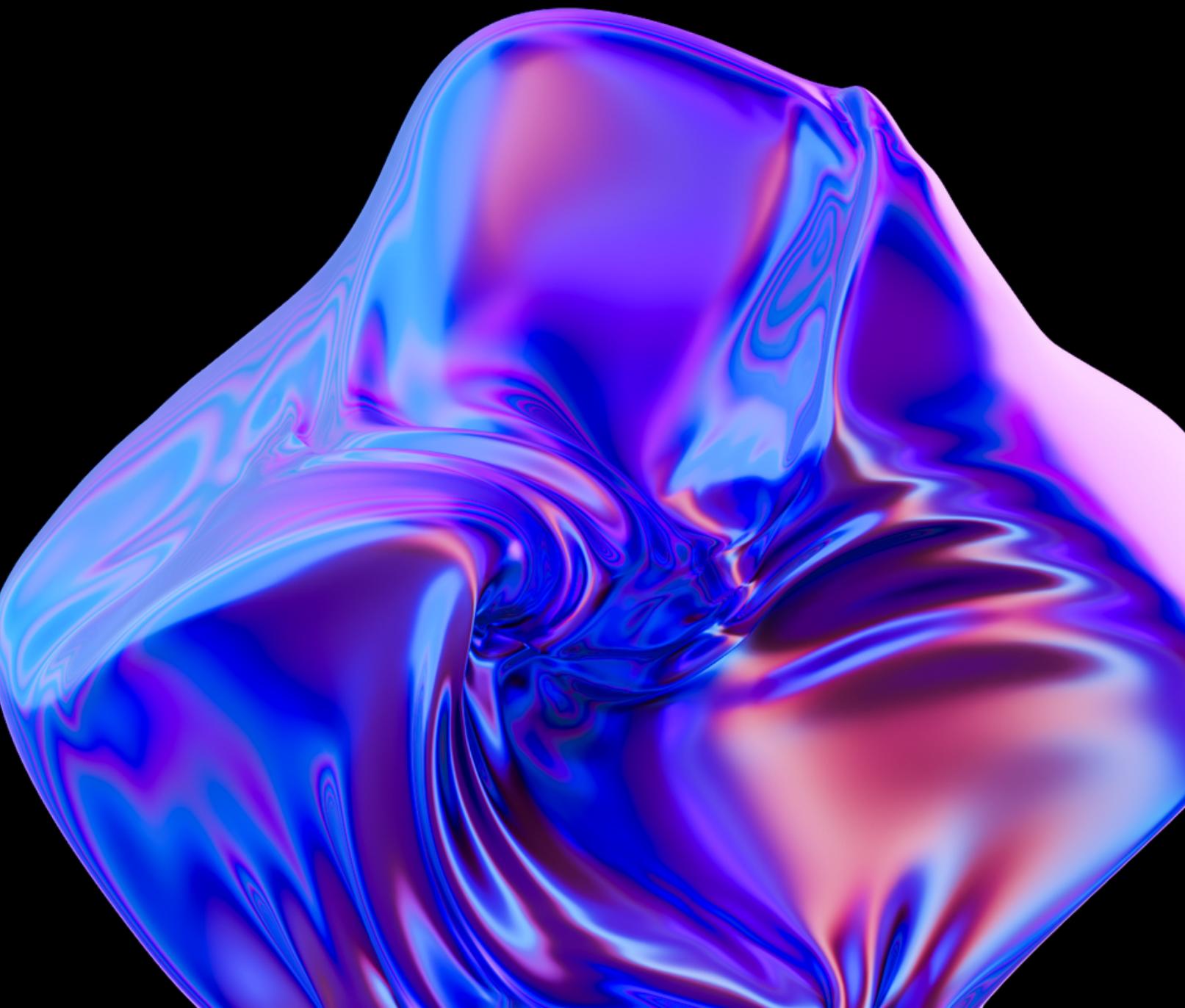


# correlation (After )



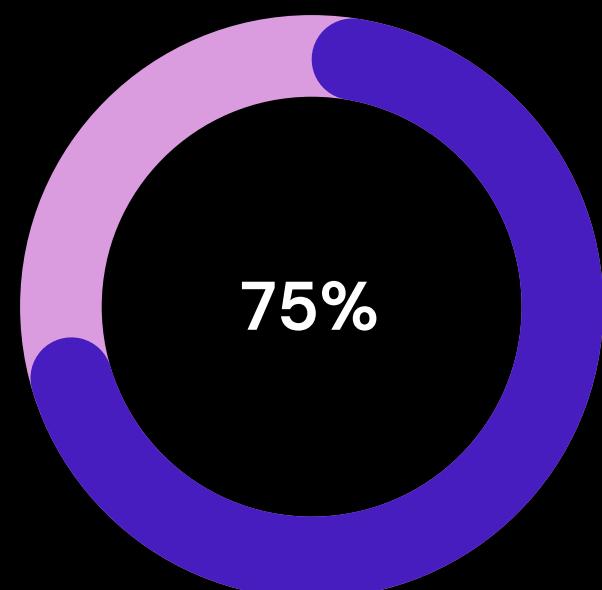
# Data After Preprocessing

- Features : 17
- Samples : 20698

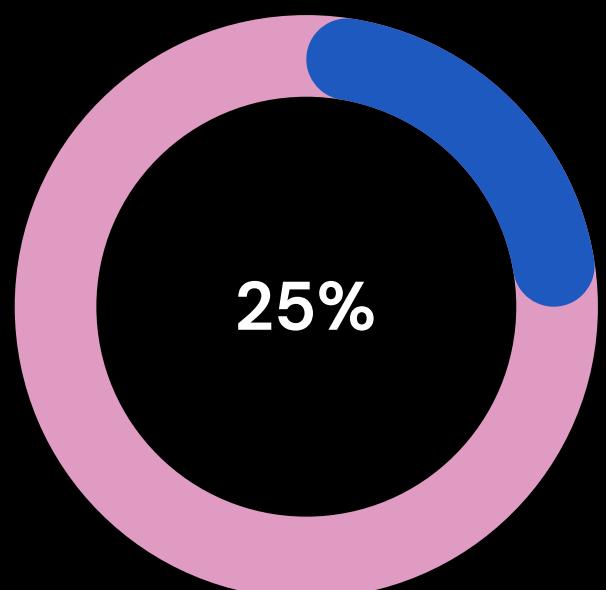


# Split Data

- we split data using 75% for train and 25% for test



Train



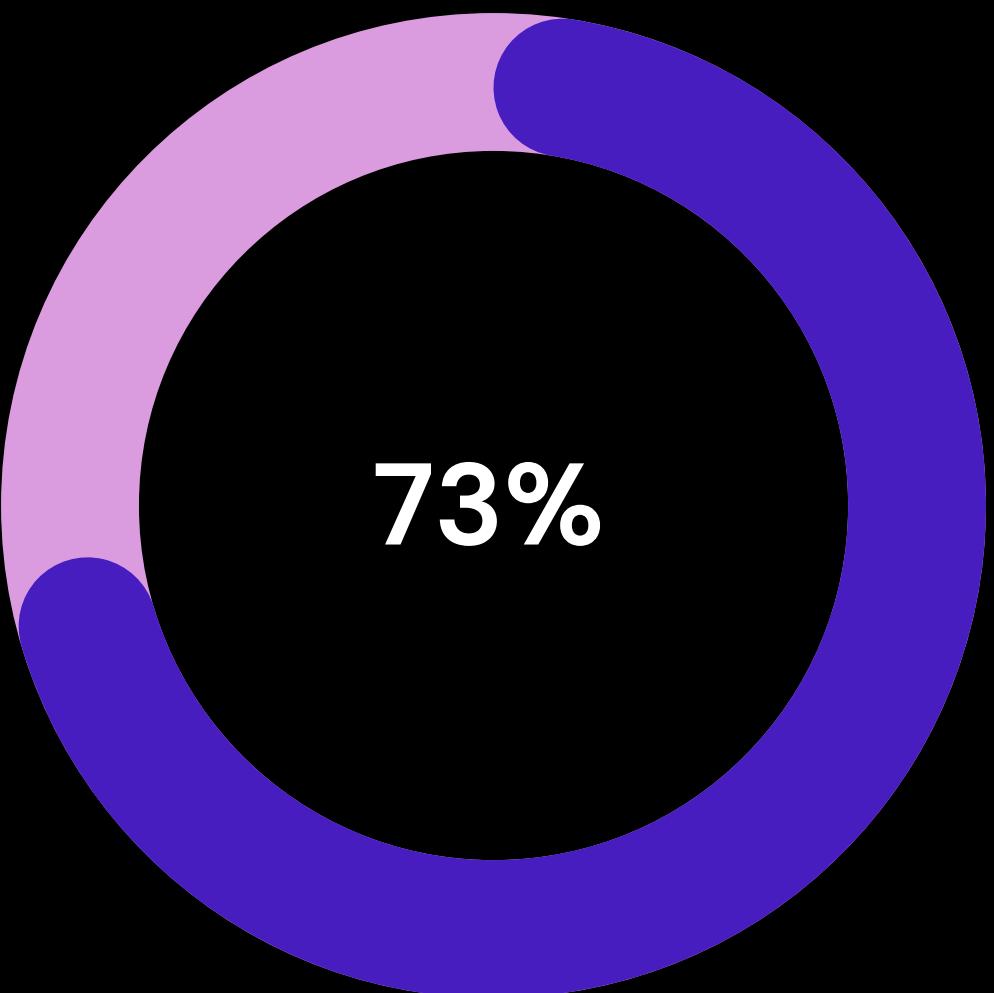
Test





# Linear Regression

- Train accuracy : 72%
- Test accuracy : 73%
- R2\_score : 73%
- mean\_squared\_error : 0.266
- mean\_absolute\_error : 0.4056

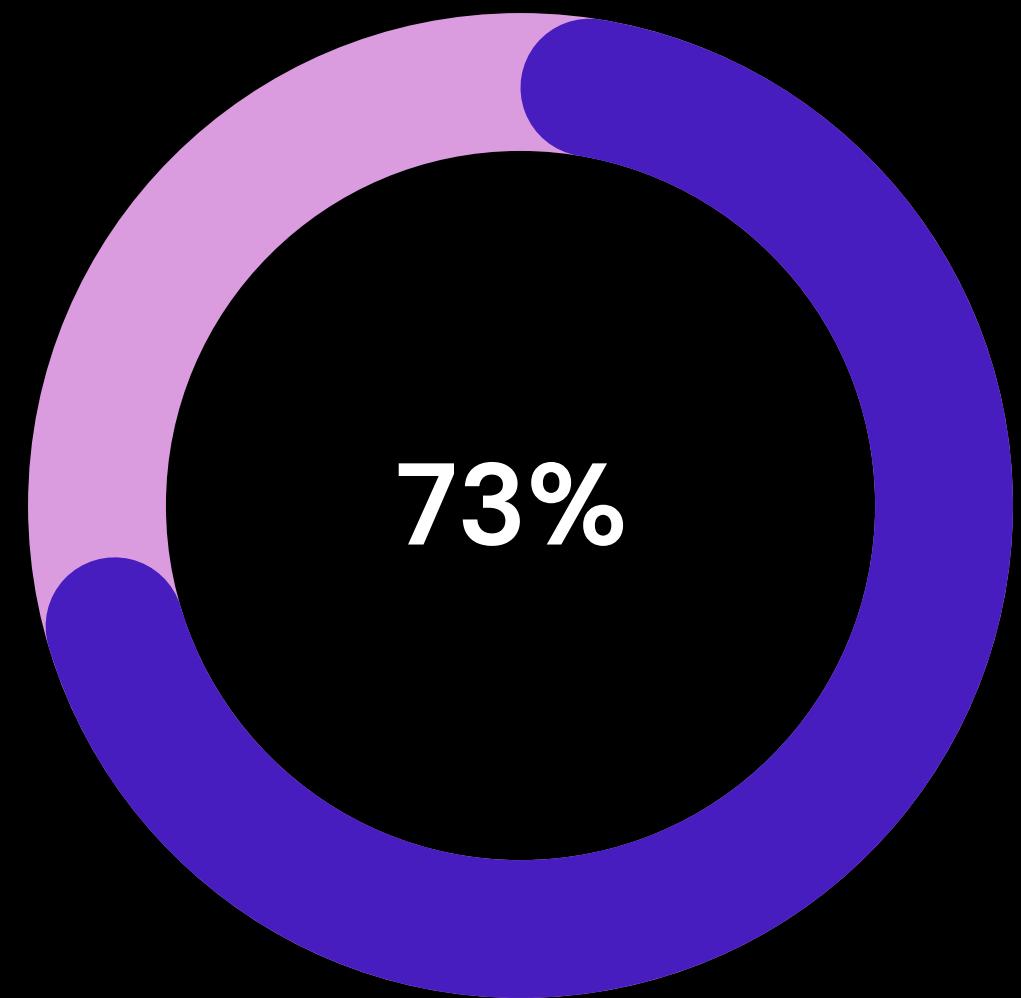




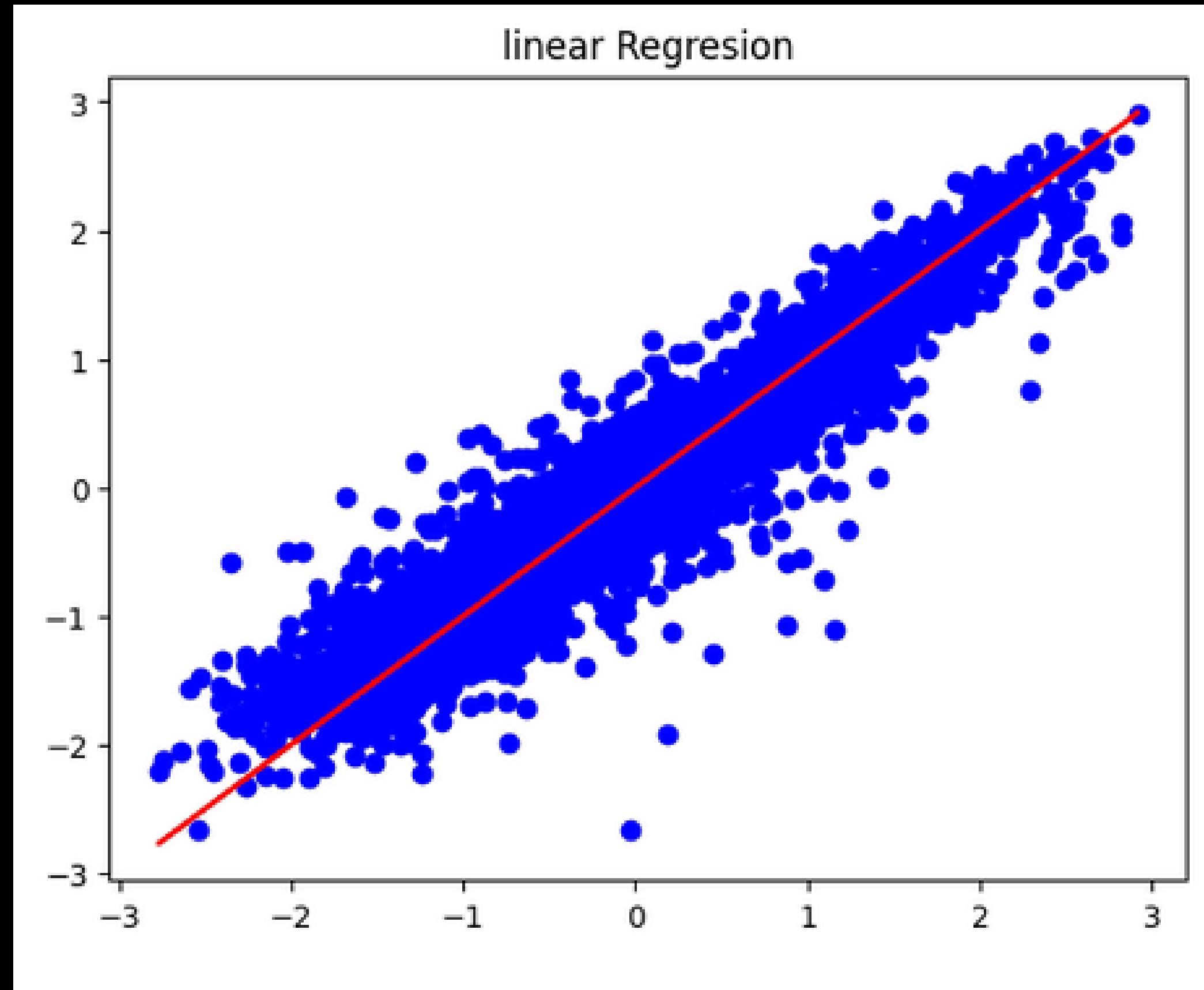
# Cross validation score

**CV = 3**

- first : 0.72400891
- second : 0.72103136
- third : 0.73081107



# Linear Regression

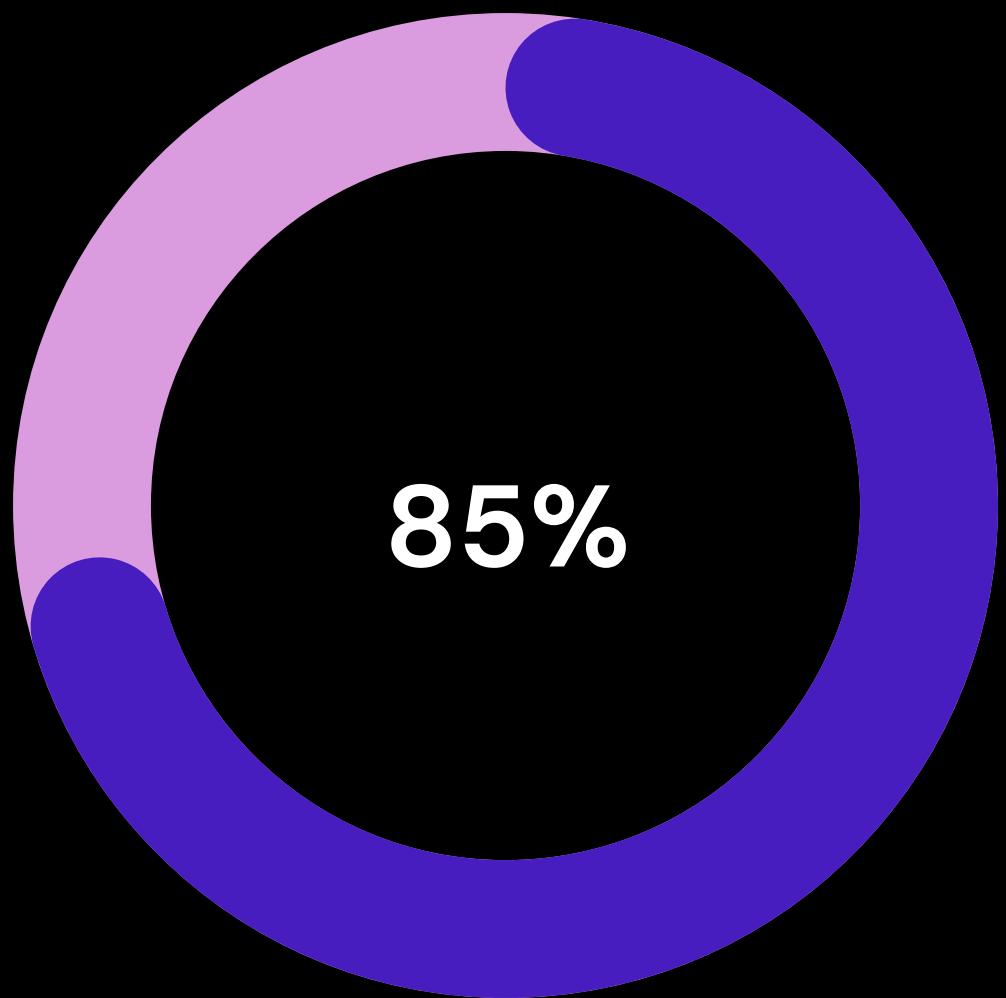




# KNeighbors Regressor

n\_neighbors=7, weights="uniform", metric="manhattan"

- Train accuracy : 88%
- Test accuracy : 85%
- R2\_score : 85%
- mean\_squared\_error : 0.142
- mean\_absolute\_error : 0.2731

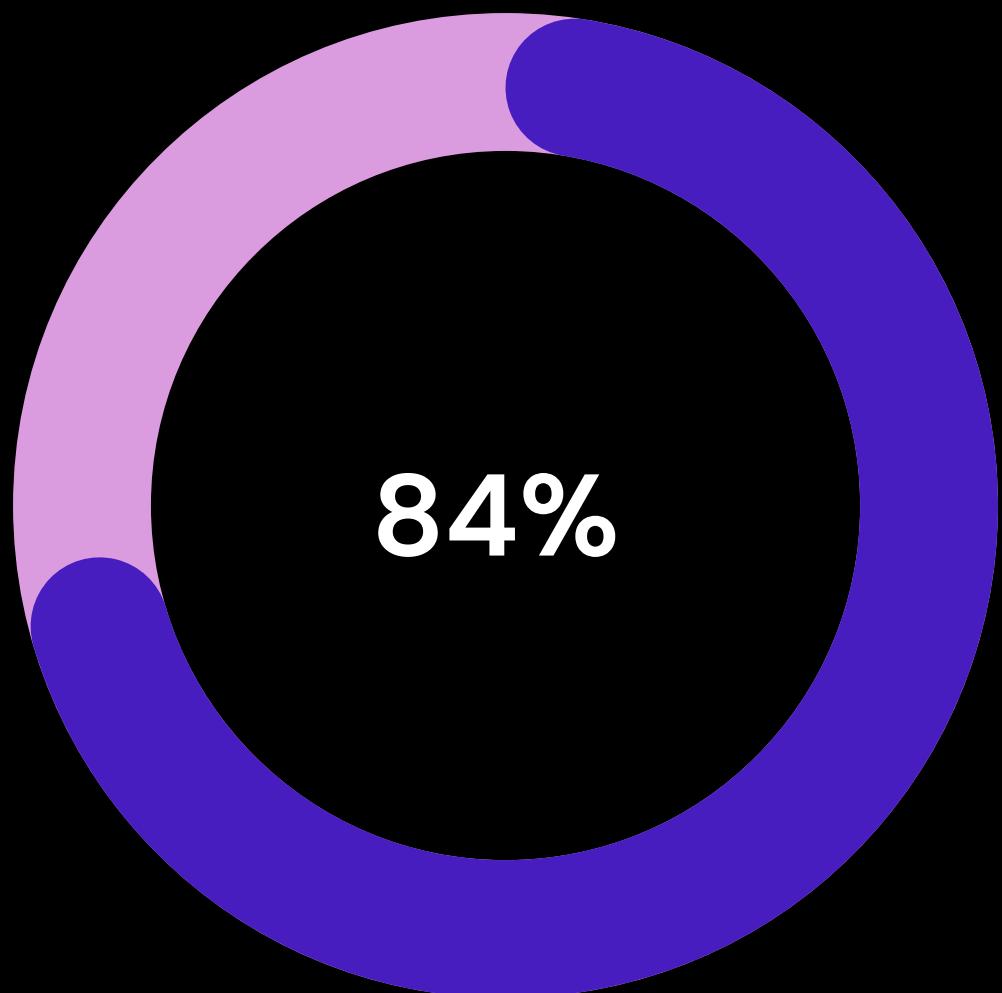


# Cross validation score

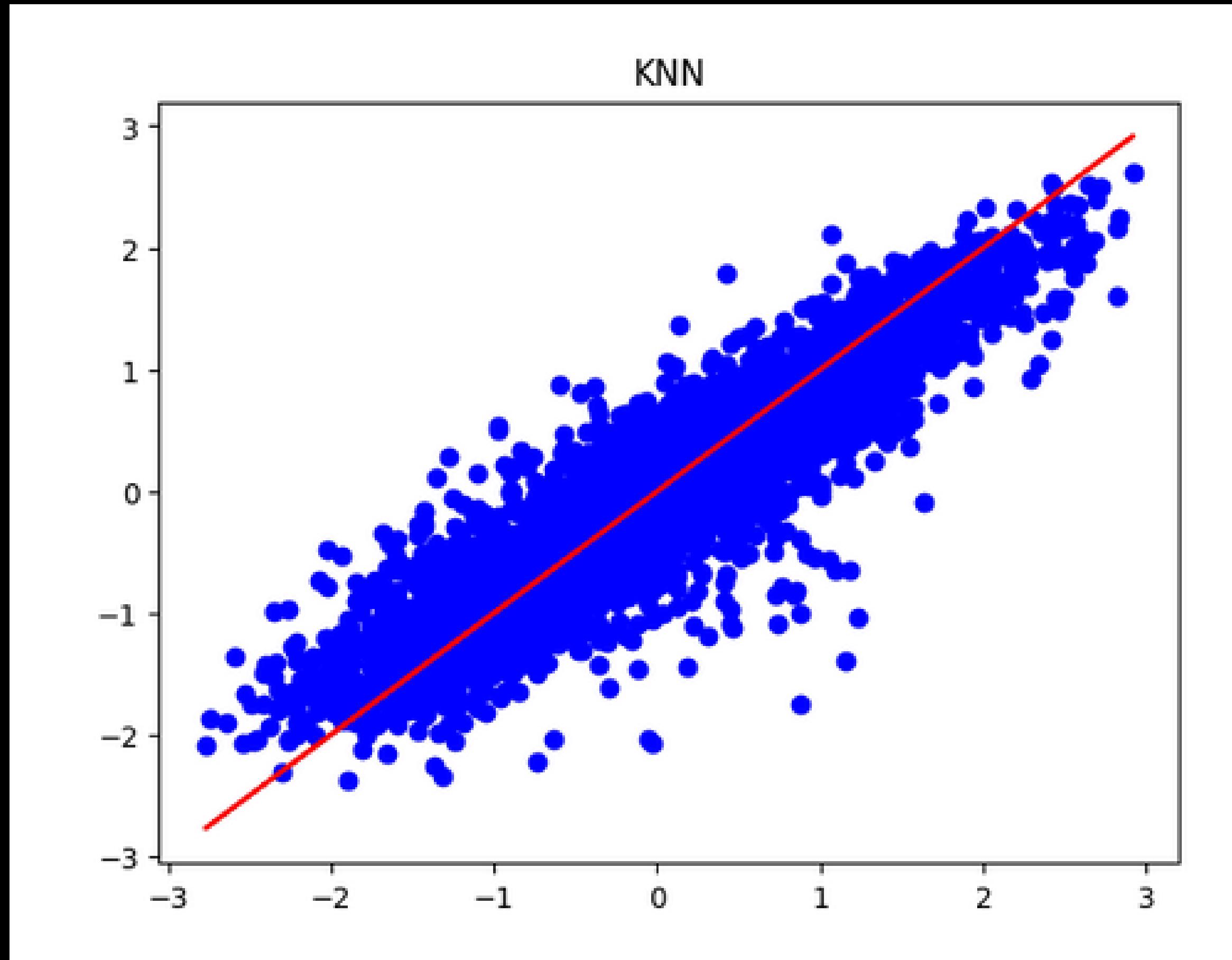


**CV = 5**

- first : 0.85222322
- second : 0.84628954
- third : 0.84869059
- forth : 0.85263857
- fifth : 0.84769341



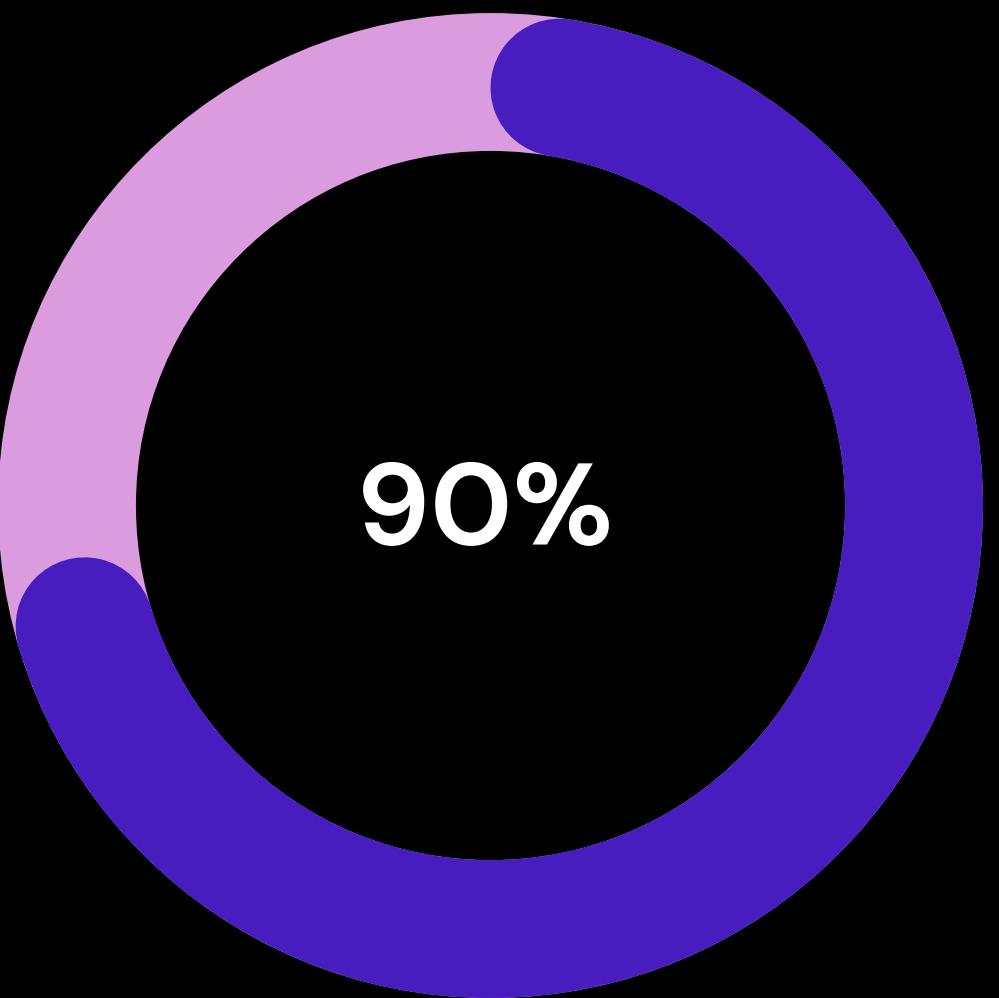
# KNeighbors Regressor





# XGBoost

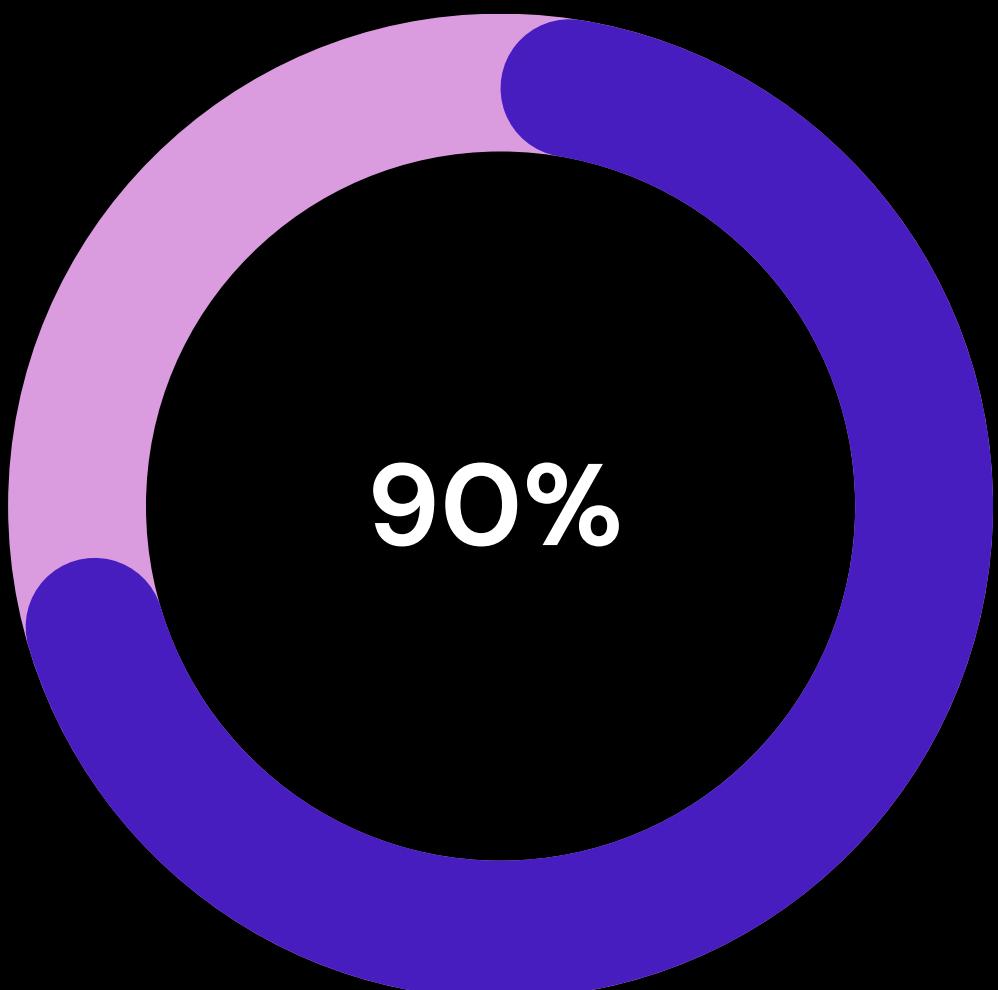
- train accuracy : 0.95596613
- test accuracy : 0.90185858
- R2\_score : 0.90
- mean\_squared\_error : 0.266
- mean\_absolute\_error : 0.4056



# Cross validation score

**CV = 3**

- first : 0.89670989
- second : 0.89310655
- third : 0.90289082

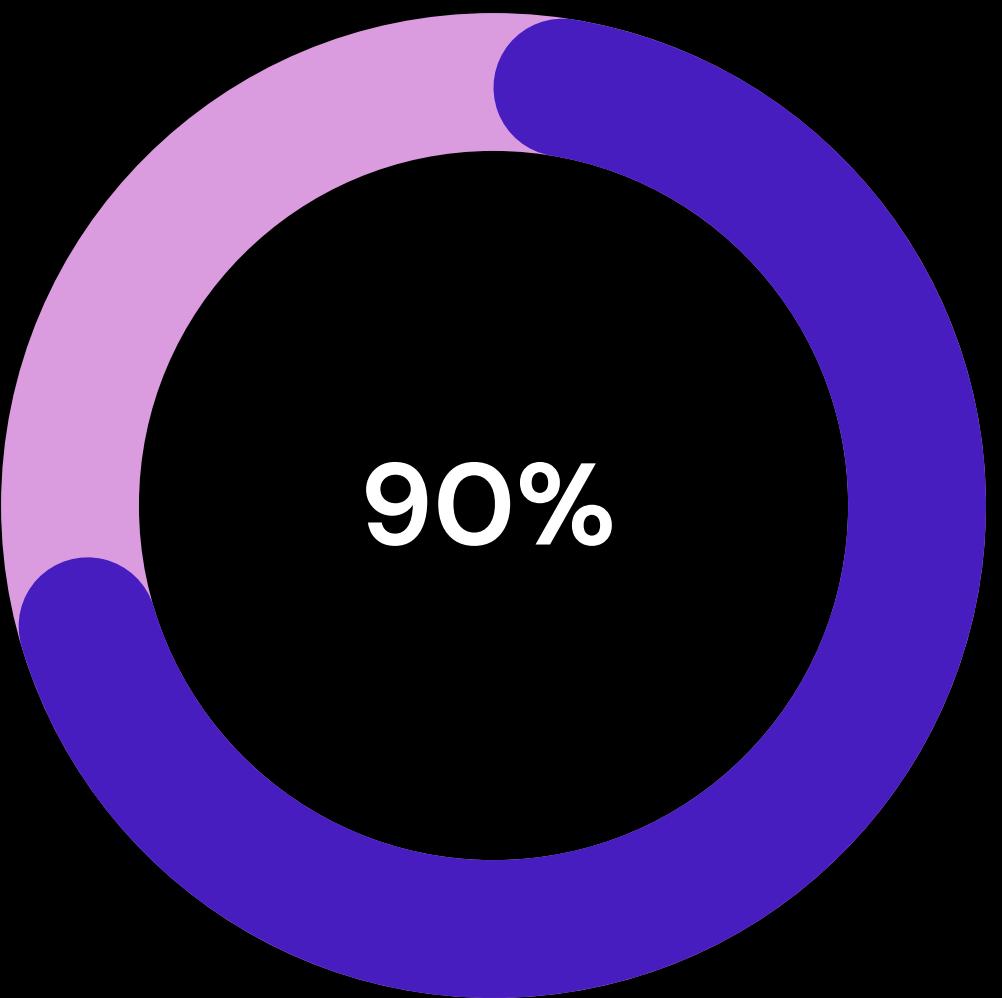




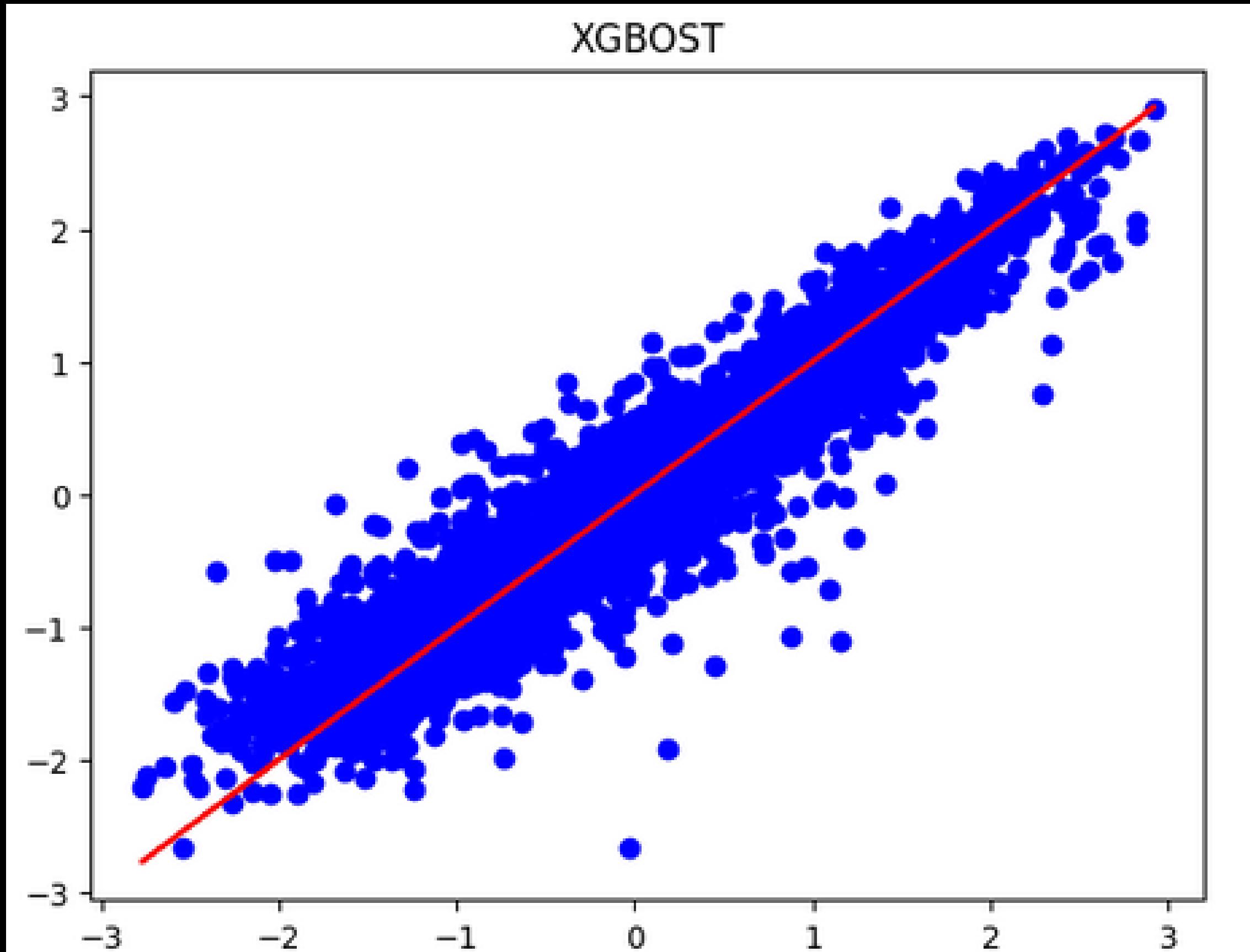
# Cross Validation

**CV=3**

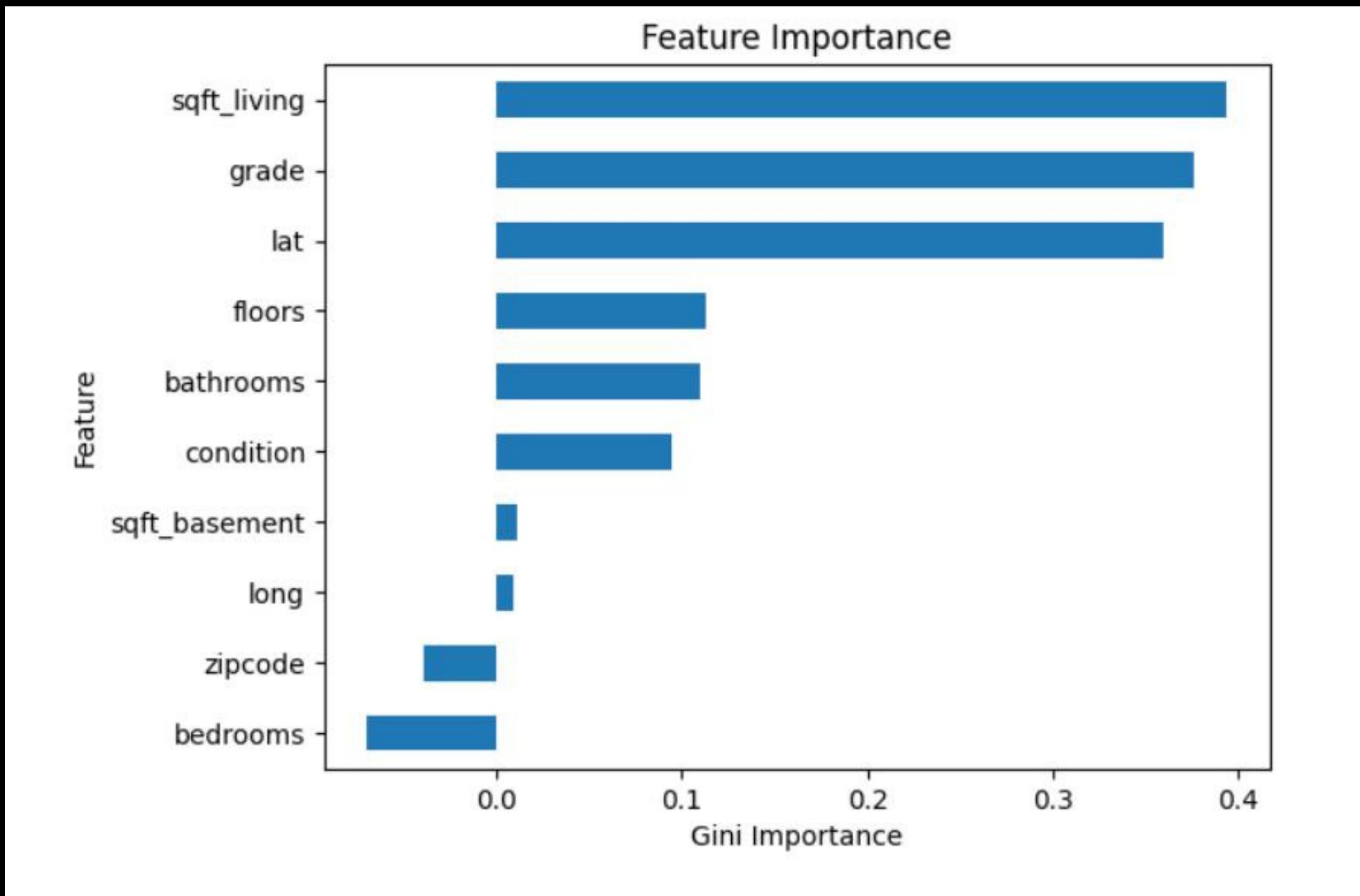
- first : 0.89670989
- second : 0.89310655
- third : 0.90289082



# XGBoost



# Feature importance





# **PlantVillage Dataset**

# Introduction

- IN THIS SECTION, WE PRESENT A MACHINE LEARNING MODEL THAT IS DEVELOPED TO DETERMINE IF THIS LEAF IS HEALTHY OR NOT

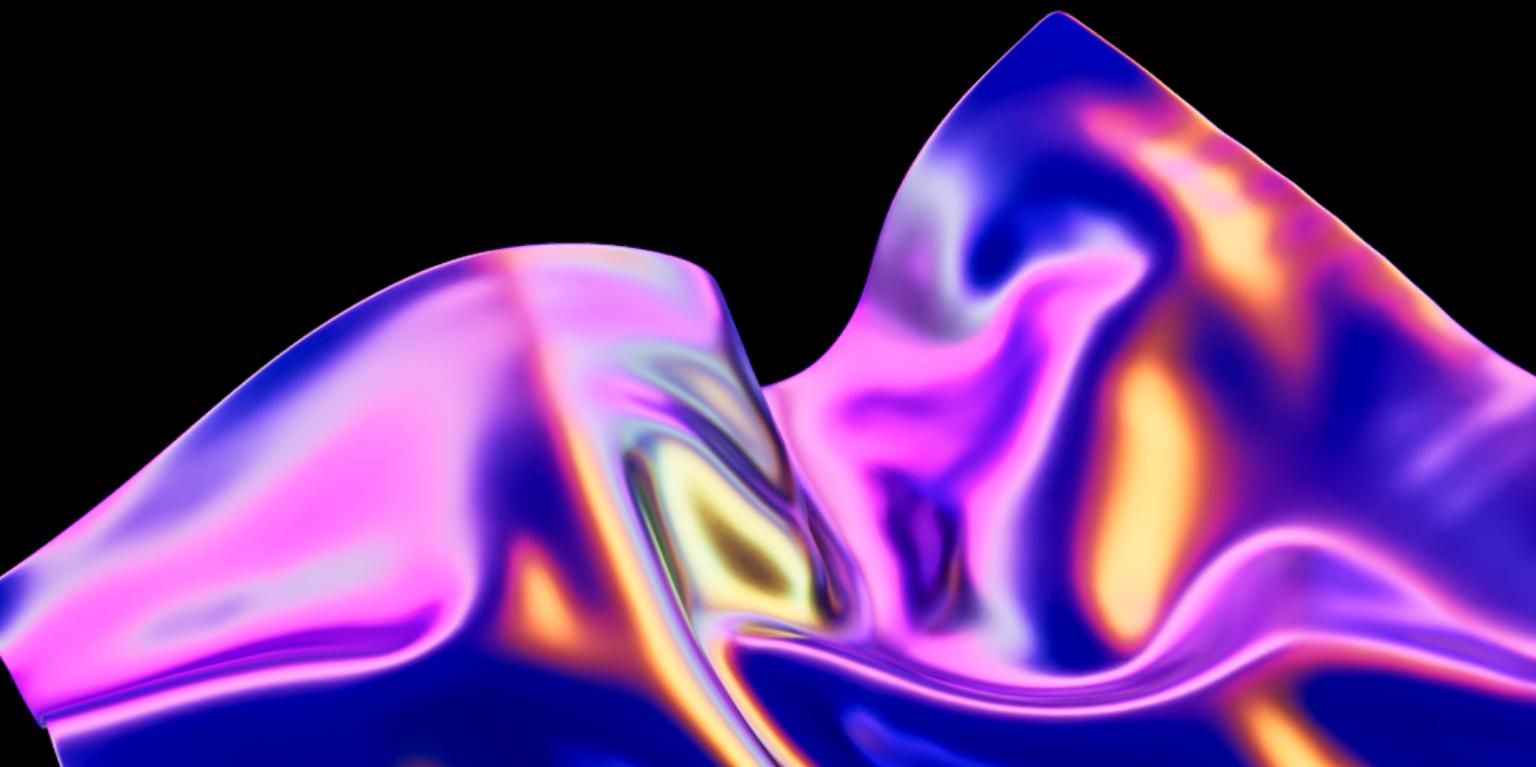
# Image DataSet

PlantVillage Dataset



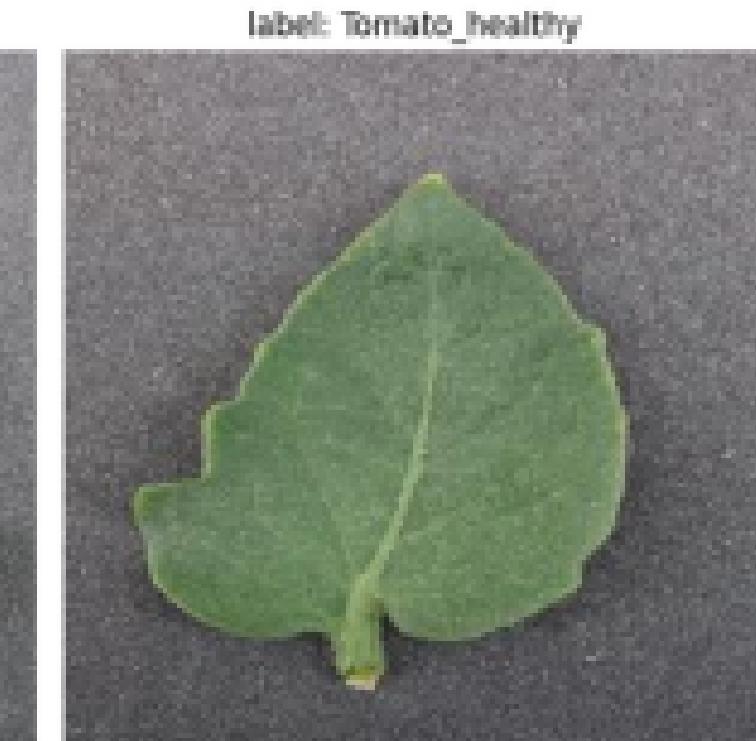
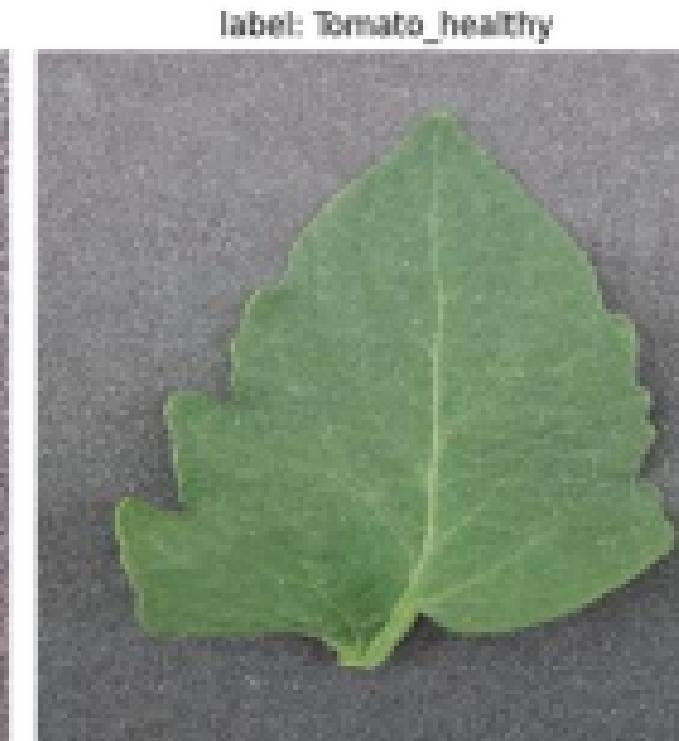
# Data Description

- Dataset of diseased plant leaf images and corresponding labels
- Data size : 690 MB
- Classes : 15 class
- image size : Width: 256, Height: 256'



# Class Selection

- 'PEPPER\_\_BELL\_\_\_BACTERIAL\_SPOT'
- 'PEPPER\_\_BELL\_\_\_HEALTHY'
- 'POTATO\_\_\_LATE\_BLIGHT'
- 'TOMATO\_\_HEALTHY'
- 'TOMATO\_LATE\_BLIGHT'





# HOGDescriptor

- Cvtcolor to convert image color from BGR TO GRAY
- use HOGDescriptor This descriptor is designed to capture the distribution of gradient orientations in an image. It divides the image into small regions called cells and computes histograms of gradient orientations within each cell.

# HOGDescriptor

- we us The compute method of the HOG descriptor takes the grayscale image as input and returns a 1D array of HOG features.
- We The flatten method is used to convert the multi-dimensional array of HOG features into a 1D array



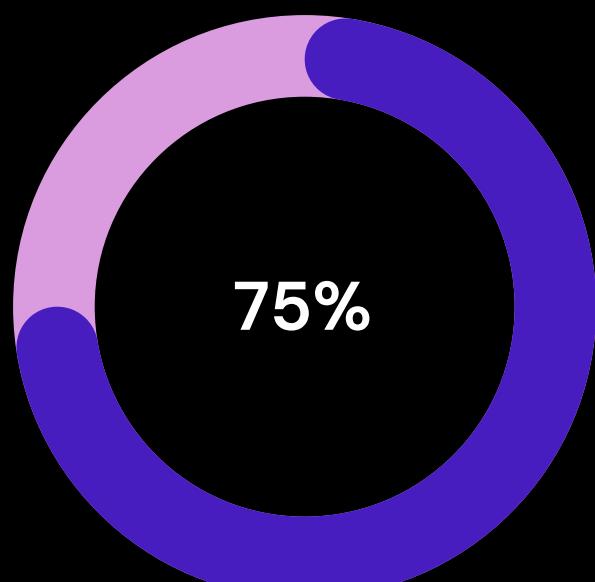
- We shuffled the data and make it as batches
- batch size is 64
- We convert the image to numpy array
- we stored images features in `features_list[]`
- we stored images labels `labels_list[]`



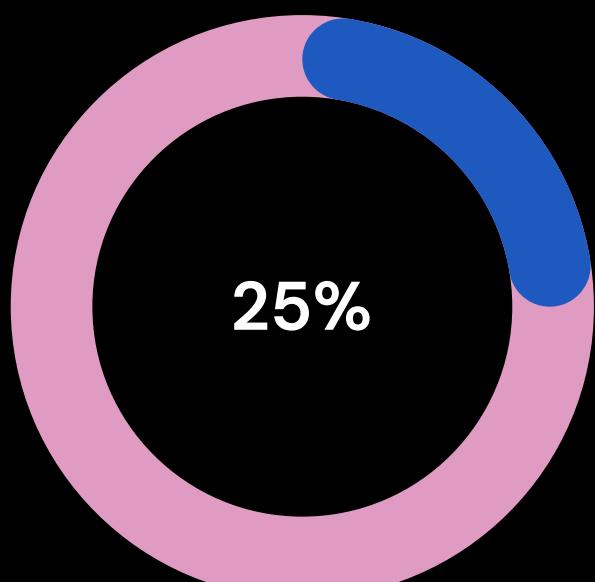
- We Convertes labels and features list to numpy array
- we used MinMaxScale() to scale features list
- we used LabelEncoder on labels list

# Split Data

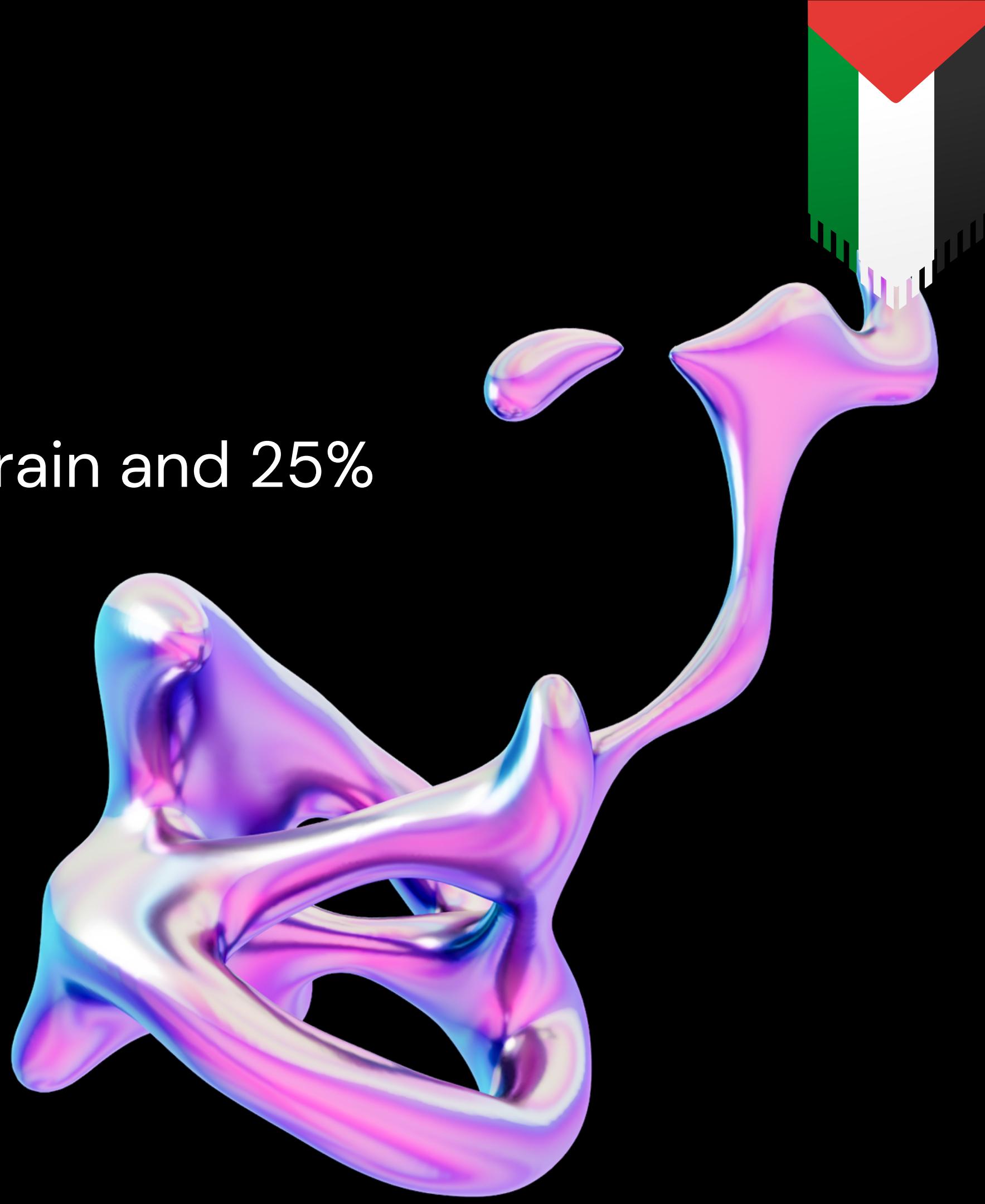
- we split data using 75% for train and 25% for test



Train



Test





# PIPELINE

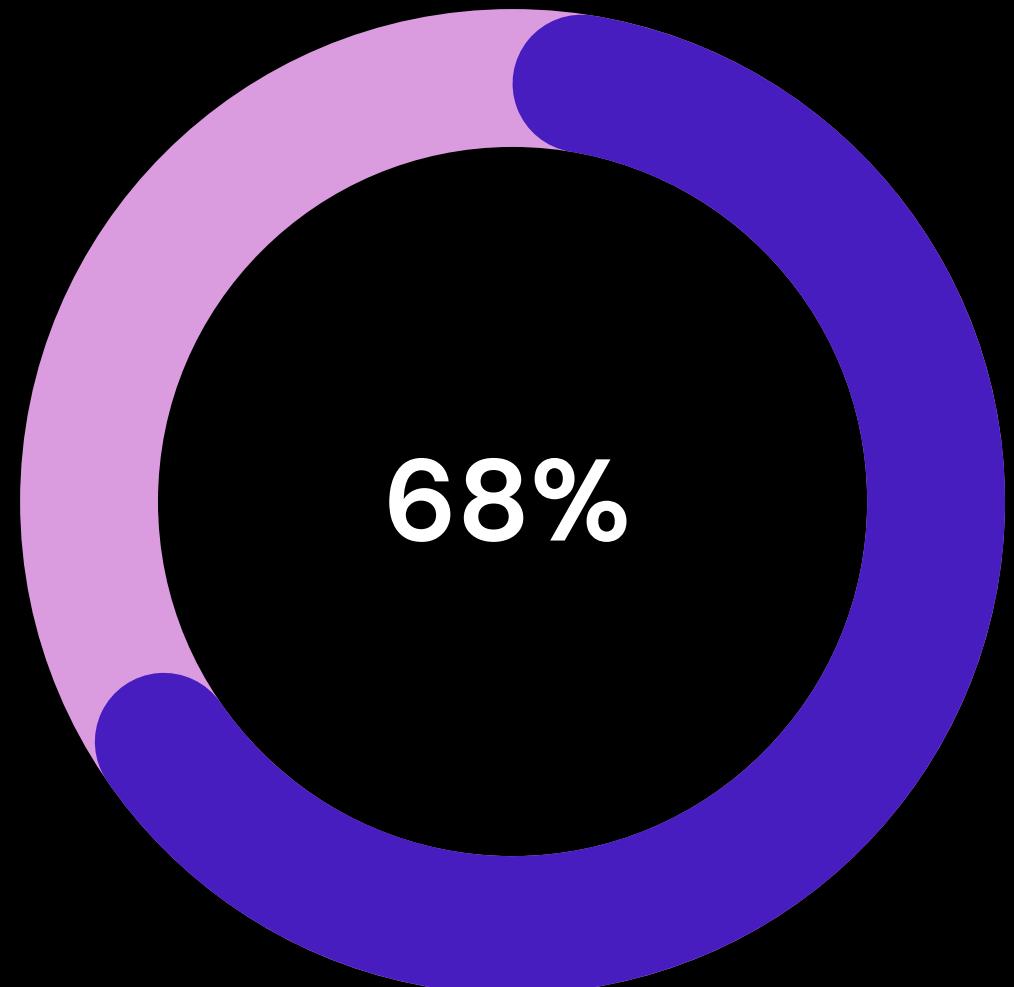
- PCA to reduce the number of features
- we changed the weights of the model , because we observed the model was biased to certain classes logistic regression
- `(max_iter=1000,  
class_weight={0: 2.5, 1: 1.5, 2: 0.8 ,3: 0.2 ,4: 0.15} ,  
random_state=42)`



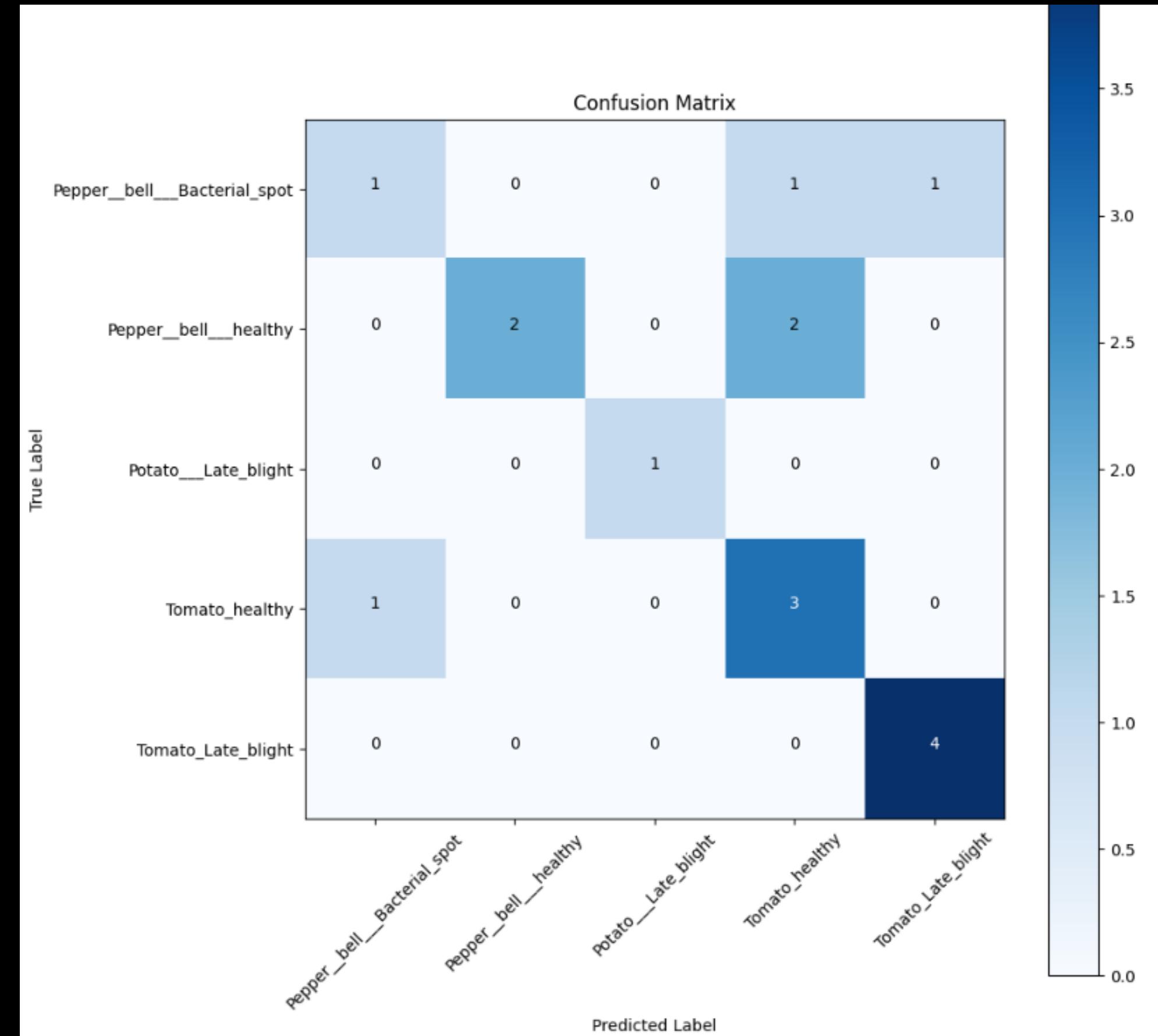
# Logistic Regression

- Accuracy : 68%

	precision	recall	f1-score	support
0	0.50000	0.333333	0.400000	3.0000
1	1.00000	0.500000	0.666667	4.0000
2	1.00000	1.000000	1.000000	1.0000
3	0.50000	0.750000	0.600000	4.0000
4	0.80000	1.000000	0.888889	4.0000
accuracy	0.68750	0.687500	0.687500	0.6875
macro avg	0.76000	0.716667	0.711111	16.0000
weighted avg	0.73125	0.687500	0.676389	16.0000



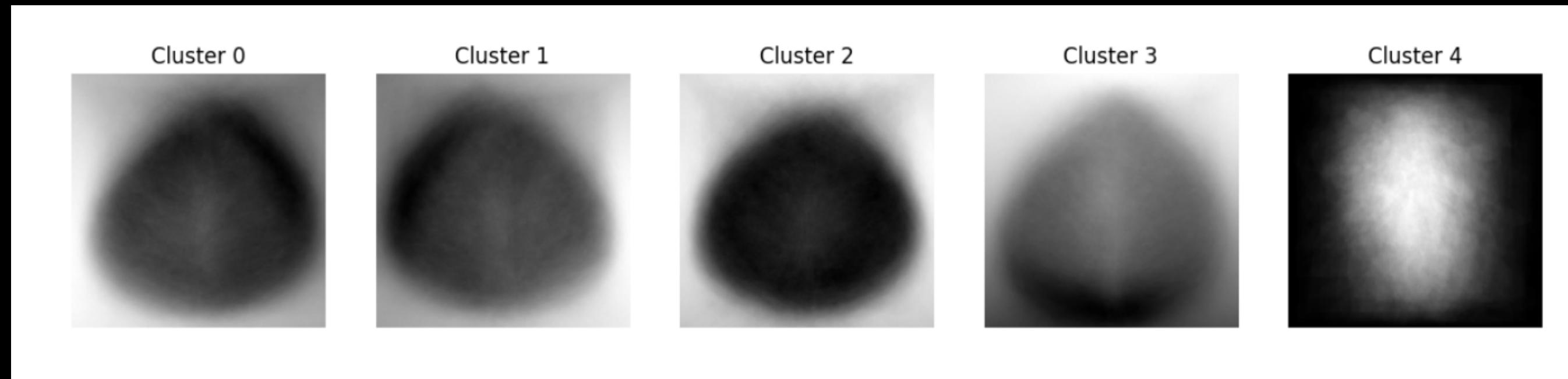
# Confusion Matrix





# K-Means Clustering

- kmeans model with hyperparameter  
`n_clusters=5, n_init= Auto`
- Centroid of each cluster  
**(combbining all images within the cluster)**





# Our team

- abdelhalim ashraf abdelhalim
- eid osama eid
- ramy ibrahim ahmed
- mohey el din maher
- ali adel sayed
- amer medhat ahmed



# Thanks