RentWise

Predicting Rental Property Prices in US

A Machine Learning Approach

Kartik Kakar

Nov 22, 2023

CONTENTS

Problem Statement

In the face of a booming rental market and soaring prices over the past decade, the increasing preference for renting over buying has resulted in a surge in rental costs, leaving many, **including myself**, grappling with the challenge of finding affordable rental properties.

The Big Question

Can we use machine learning to accurately predict the Rental Property prices in US?

My Solution

RentWise

A robust Predictive model that accurately estimates rental prices.

Value Proposition

Improved decision making and insights for:

- Renters
- Landlords
- The ever evolving Real Estate industry

Dataset Overview

Source: Kaggle

Rows: 384,977
Columns: 22

Unique Data entries

No Duplicates

Integers

- sqfeet
- beds
- cats/dogs/smoking
- wheelchair access
- comes furnished
- electric vehicle charge

Floats

- latitude
- longitude
- baths

Strings

- region
- type
- laundry option
- parking option
- desription
- state

Target Column

- Price

Project Workflow

Data Cleaning

- Dealt with missing values.
- Dropped unwanted columns.

EDA

- Dealt with outliers.
- Visualized distributions of features.
- Check for collinearity.

Data Preprocessing

- Analyzed categorical columns.
- One-hot encoded.
- Combined all numerical columns.
- Train-Test split (80/20).

Baseline modelling

- Tried different regression models.
- Scaled the dataset.
- Compared all models on scaled data.

Hyperparameter tuning

- Tried different hyperparameters.
- Compared accuracies.
- Found best parameters
- Evaluated model on Test set.
- Made Predictions.

Final Steps

- Training the model on entire dataset.
- Make an interactive website.
- Make a Tableau Dashboard.

Baseline Modelling

A

Logistic Regression

B

Lasso Regression

C

Ridge Regression

D

Decision Tree Regressor

E

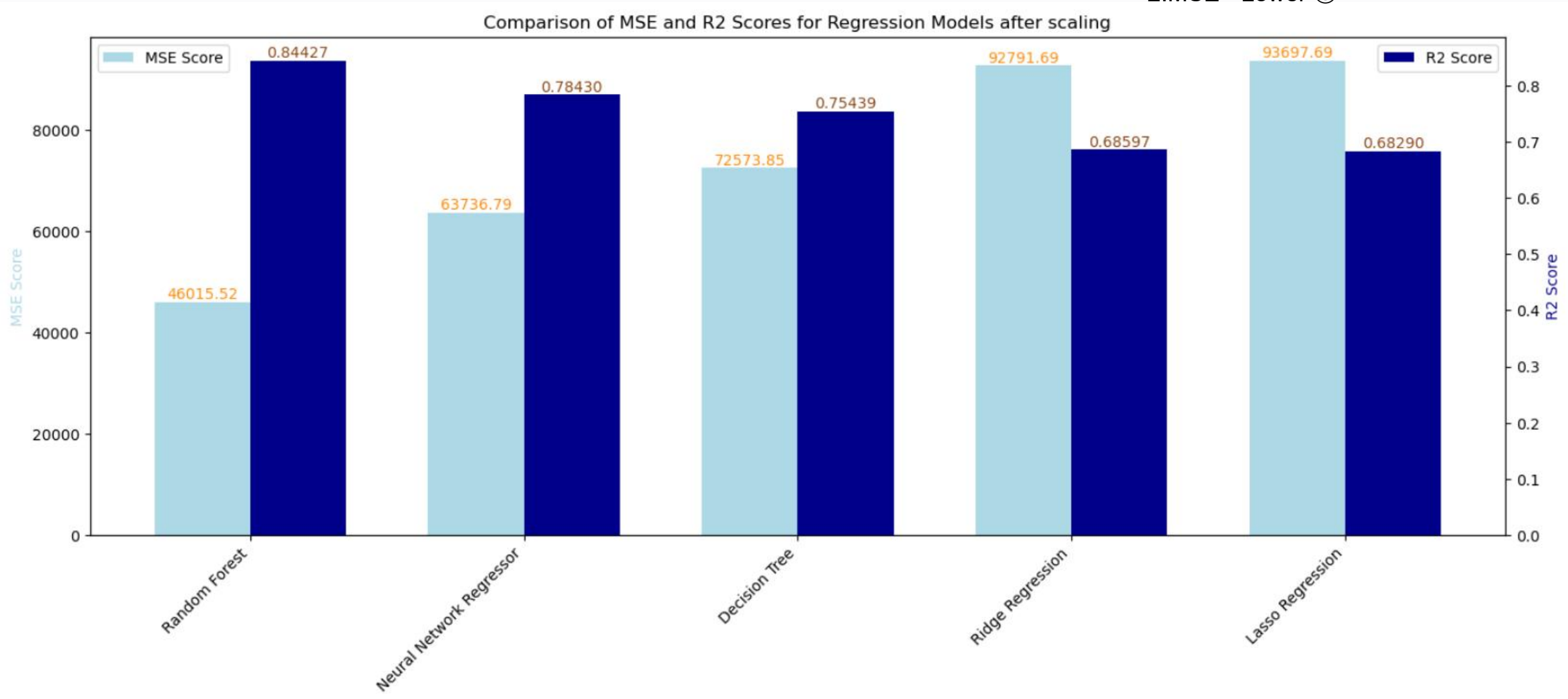
Random Forest Regressor

F

Neural Network Regressor

Model Comparison

Metrics used:
1.R2 score - Higher 😊
2.MSE - Lower 😊



Hyperparameter tuning - Random Forest Regressor

1 Sampled - Split - Scale

- Took a sample of 50,000 rows.
- For reducing the computing time of our model.
- Split to Train (80%) and Test (20%) set.
- Used Standard Scaler.

2 PCA

- Applied PCA for generating uncorrelated principal components.
- Chose a threshhold of 0.9 to retain 90% variance.
- **Optimal no. of components: 380**

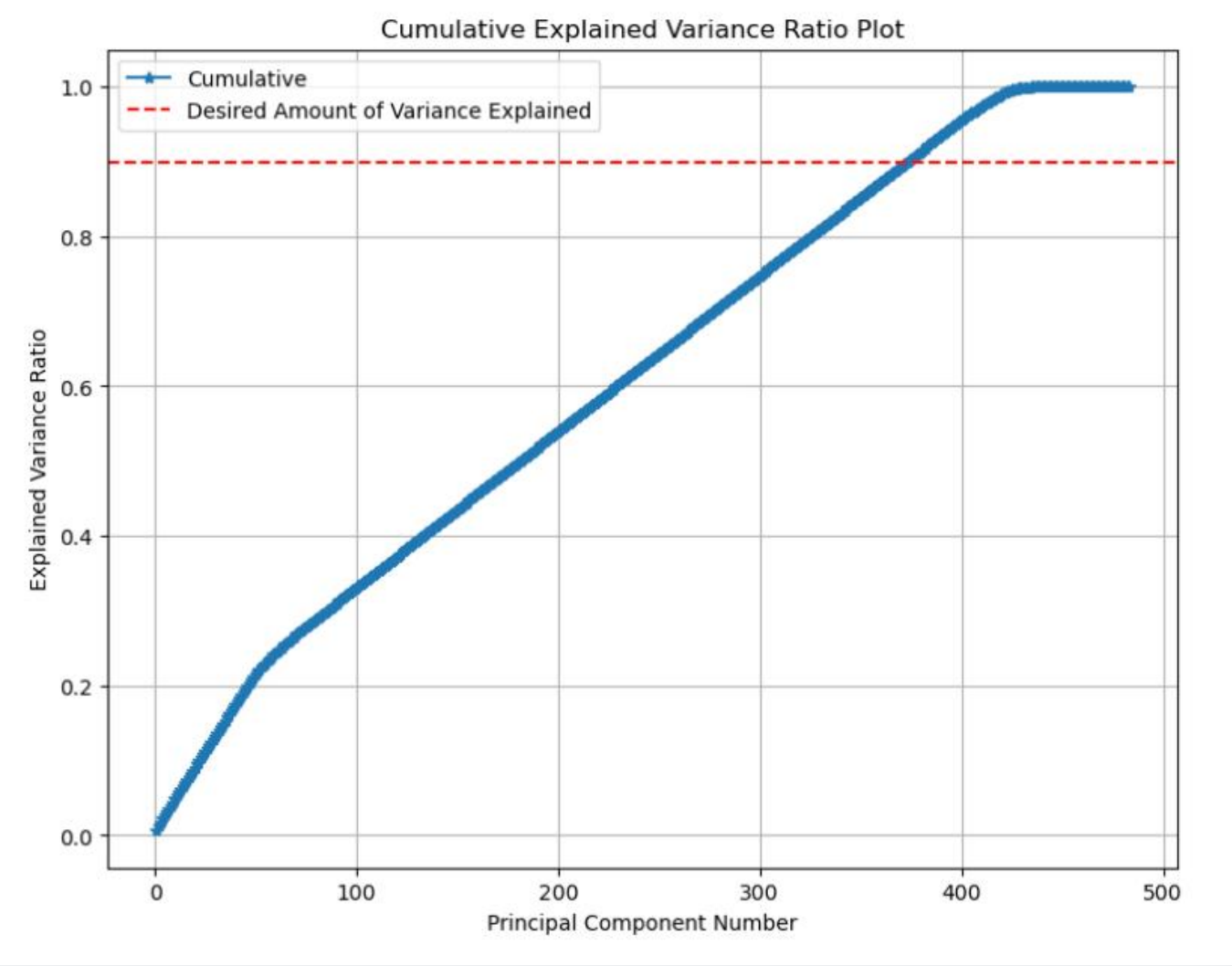
3 Maximum Depth

- Initial range: 50 to 151, step size of 25
- Optimal initial max depth: 150
- Upon further tuning, **Optimal max depth: 140**

4 Min. sample leaf, min. sample split

- Created a pipeline, used gridsearch.
- Best hyperparameters:
  - **Optimal min. sample leaf: 5**
  - **Optimal min. sample split: 0.01**

Principle Component Analysis



Variance threshold: 90%

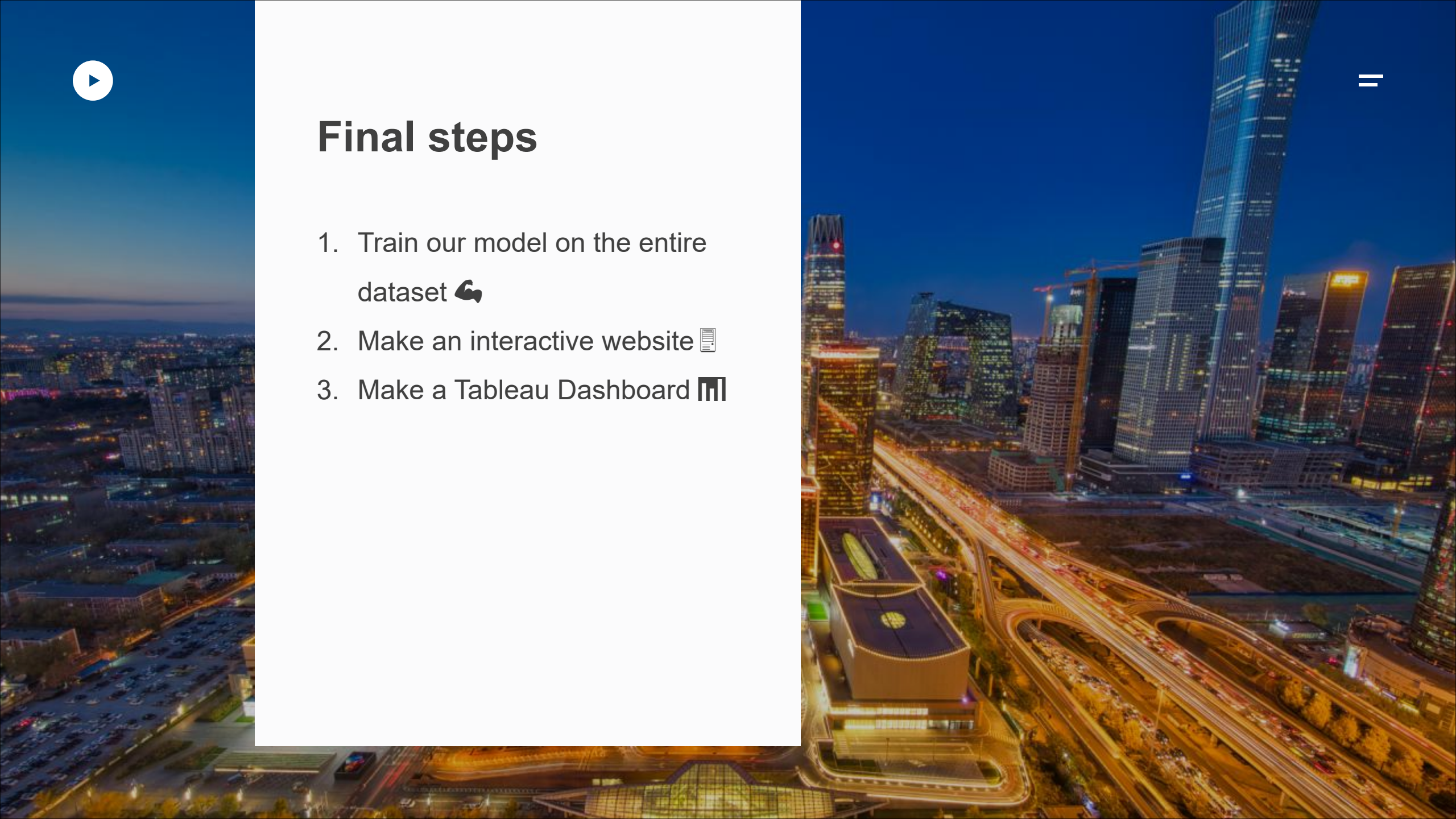
Optimal n_components: 380

Model Evaluation Examples

#	Region	Type	SqFeet	Beds	Baths	Cats	Dogs	Smoking	Wheel chair	Electric Vehile Charge	Furnished	Laundry	Parking	State	Price
1	reno / tahoe	apartment	1078	3	2	1	1	0	0	0	0	w/d in unit	carport	CA	**$2850.08**
2	reno / tahoe	condo	1001	2	2	0	0	0	0	0	0	w/d hookups	carport	CA	**$2510.96**
3	reno / tahoe	apartment	1944	3	3	1	1	1	0	0	0	w/d in unit	attached garage	CA	**$3119.40**

Final steps

1. Train our model on the entire dataset 💪
2. Make an interactive website 🖥
3. Make a Tableau Dashboard 📊



THANKS