Final Presentation

Rents Prediction Model Based on House Attributes and Public Transportation Accessibility

Group 18: Yijing Zhou & Jiachen Huang

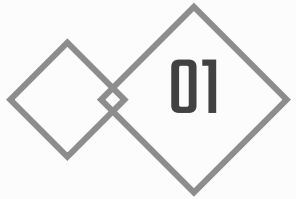
AGENDA

1 Introduction

2 Data & Feature

3 Methodology

4 Result & Future Work



Introduction

Project Goal:

Develop a model that can predict the rent, given house attributes and the accessibility to public transportations.

- Predict the fair rent of an apartment
- Suggest on public transportation planning
- Find undervalued units

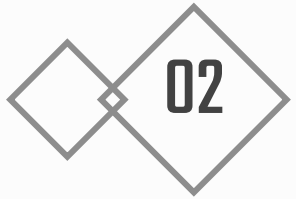
01 Introduction

Significance:

9.6 million migrants in Shanghai, 80% rent their house

Helped assess fair rent price and find undervalued units

Source: Sadayuki, Taisuke. “Measuring the Spatial Effect of Multiple Sites: An Application to Housing Rent and Public Transportation in Tokyo, Japan.” *Regional Science and Urban Economics*, vol. 70, 2018, pp. 155–173., doi:10.1016/j.regsciurbeco.2018.03.002.



Data & Features

Data Source:

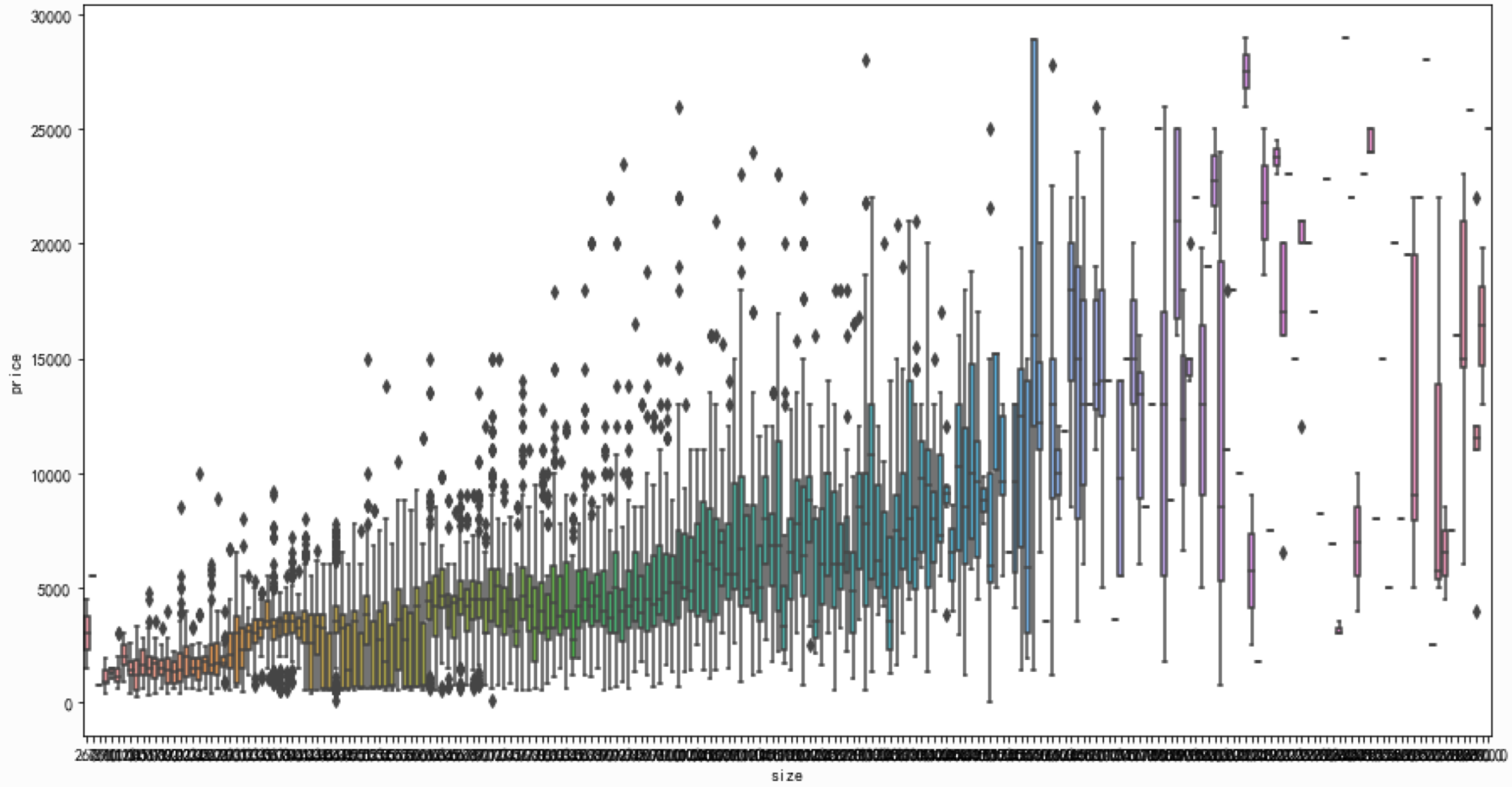
ganji.com

Web Crawler to download, regular expression and get dummies to process

MetrodataTeam's city database

Import ArcGis data and transform into public transportation attributes

02 Data & Features



02 Data & Features

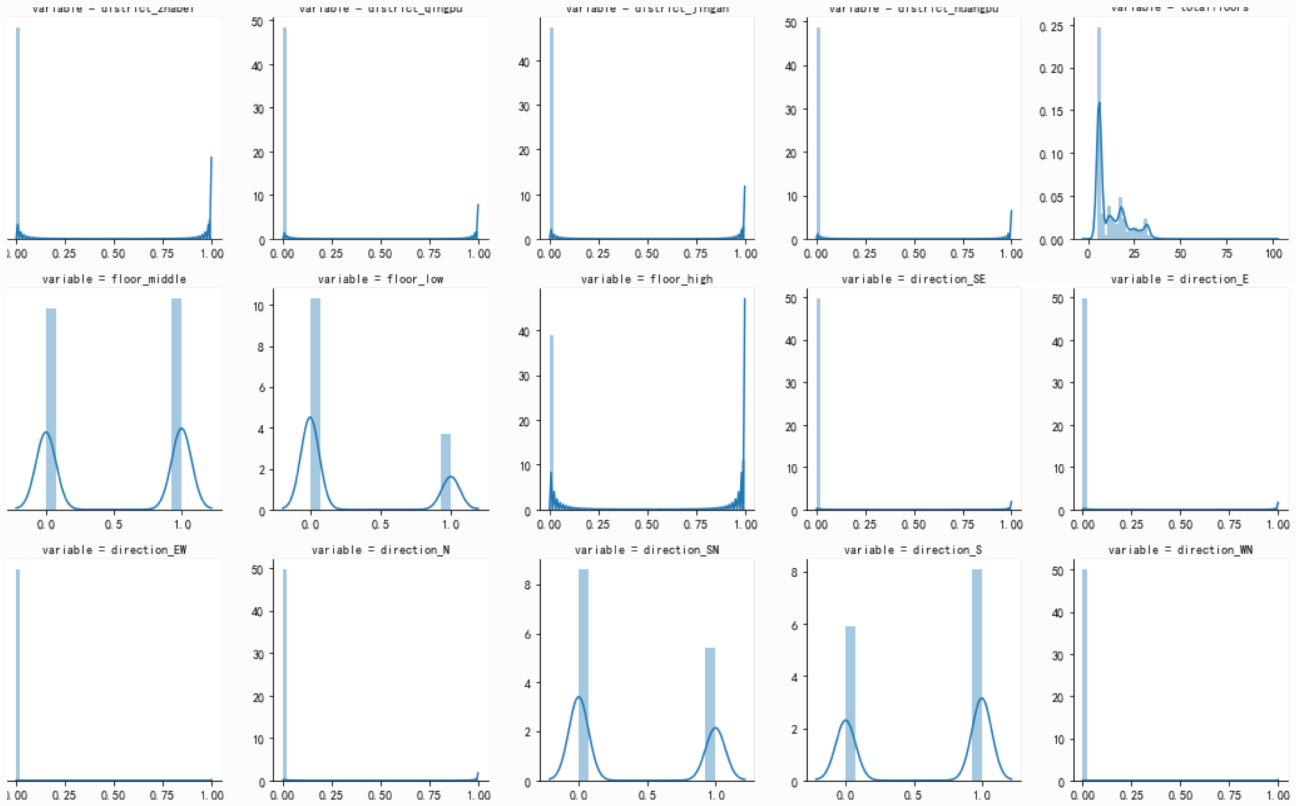
Features:

Size, Room type, Payment, District, Floors, Direction, Facilities, Decoration, Bus and Metro stations nearby...

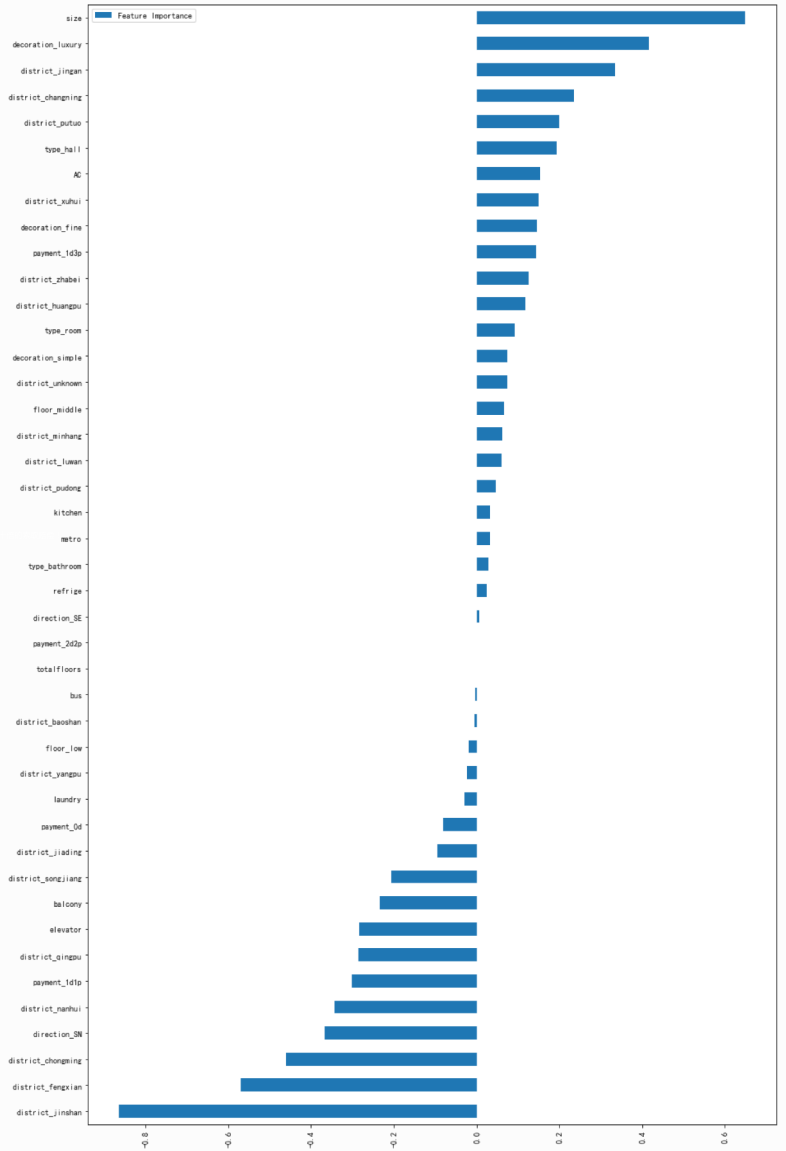
```
In [654]: df.columns

Out[654]: Index(['price', 'size', 'type_room', 'type_hall', 'type_bathroom',
                 'payment_1d1p', 'payment_1d3p', 'payment_1d2p', 'payment_2d1p',
                 'payment_2d3p', 'payment_2d2p', 'payment_0d', 'elevator',
                 'district_nanhui', 'district_luwan', 'district_jiading',
                 'district_fengxian', 'district_baoshan', 'district_chongming',
                 'district_xuhui', 'district_putuo', 'district_unknown',
                 'district_yangpu', 'district_songjiang', 'district_pudong',
                 'district_hongkou', 'district_jinshan', 'district_changning',
                 'district_minhang', 'district_zhabei', 'district_qingpu',
                 'district_jingan', 'district_huangpu', 'totalfloors', 'floor_middle',
                 'floor_low', 'floor_high', 'direction_SE', 'direction_E', 'direction_N',
                 'direction_SN', 'direction_S', 'direction_WS', 'direction_W',
                 'decoration_simple', 'decoration_fine', 'decoration_luxury', 'metro',
                 'bus', 'balcony', 'refrige', 'laundry', 'kitchen', 'AC'],
                dtype='object')
```

02 Data & Features



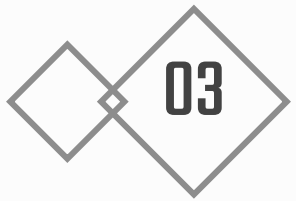
Skewness & Logarithm



LASSO

03 Methodology

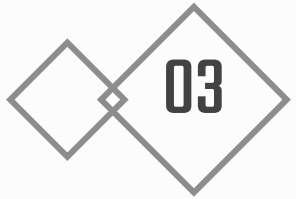
1. Baseline and Evaluation Metric
2. Regression Models
3. Hyperparameters Tuning
4. Ensemble Methods



Baseline and Evaluation Metric

Baseline: Linear Regression

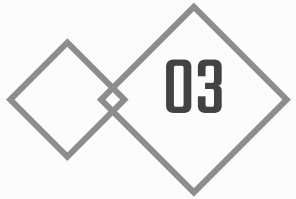
Evaluation Metric: Rooted Mean Square Error (RMSE)



Regression Models

In [508]:
```
#get result of 13 default models
names = ["LR", "Ridge", "Lasso", "RF", "GBR", "SVR", "LinSVR", "Ela","SGD","Bay","Ker","Extra","Xgb"]
print("   mse_train" + " mse_std" + " mse_test")
for name, model in zip(names, models):
    score = rmse_cv(model, X_train, Y_train)
    model.fit(X_train,Y_train.ravel())
    Y_hat = model.predict(X_test)
    rmse_test = np.sqrt(mean_squared_error(Y_test,Y_hat))
    print("{}: {:.6f}, {:.4f}, {:.4f}".format(name,score.mean(),score.std(),rmse_test))
```

```
   mse_train mse_std mse_test
LR: 0.475950, 0.0149, 0.4790
Ridge: 0.475896, 0.0149, 0.4789
Lasso: 0.511728, 0.0165, 0.5147
RF: 0.273322, 0.0214, 0.2533
GBR: 0.340178, 0.0150, 0.3432
SVR: 0.344476, 0.0275, 0.3471
LinSVR: 0.529947, 0.0135, 0.5378
Ela: 0.476835, 0.0151, 0.4805
SGD: 0.484469, 0.0157, 0.4842
Bay: 0.475883, 0.0149, 0.4789
Ker: 0.366681, 0.0215, 0.3614
Extra: 0.248990, 0.0055, 0.2491
Xgb: 0.343567, 0.0177, 0.3470
```



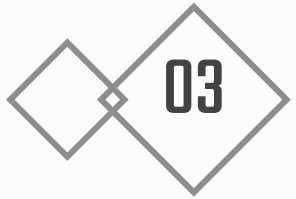
Regression Models

Model	Training error	Testing error
Linear Regression	0.475950	0.4790
Ridge	0.475896	0.4789
Lasso	0.511728	0.5147
Random Forest	0.273322	0.2533
Gradient Boost	0.340178	0.3432
SVR	0.344476	0.3471
Linear SVR	0.529947	0.5378
Elastic Net	0.476835	0.4805
SGD	0.484469	0.4842
Bayesian Ridge	0.475883	0.4789
Kernel Ridge	0.366681	0.3614
Extra	0.248990	0.2491
XGB	0.343567	0.3470

03 Hyperparameters Tuning

Model	Training error	Testing error
Random Forest	0.2350	0.2317
Gradient Boost	0.2395	0.2354
SVR	0.2766	0.2614
Kernel Ridge	0.3174	0.3271
Extra	0.2448	0.2378
XGB	0.2225	0.2256

Grid Search VS. Random Search...



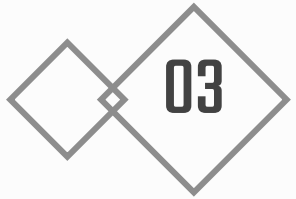
Hyperparameters Tuning

Kernal Ridge

In [323]:
```
#Kernal random search
param_random={'alpha':[0.01,0.05,0.1,0.2,0.3,0.5], 'kernel':["polynomial"], 'degree':[2,3,4],'coef0':[0.8,1,1.2,1.5,2,2
random_search(KernelRidge()).random_get(X_train,Y_train,X_test,Y_test,param_random)
```

```
{'kernel': 'polynomial', 'degree': 3, 'coef0': 2, 'alpha': 0.1} 0.3173723097695623
                                              params  mean_test_score
0  {'kernel': 'polynomial', 'degree': 4, 'coef0':...         0.411458
1  {'kernel': 'polynomial', 'degree': 4, 'coef0':...         0.337013
2  {'kernel': 'polynomial', 'degree': 4, 'coef0':...         2.661175
3  {'kernel': 'polynomial', 'degree': 4, 'coef0':...         0.327846
4  {'kernel': 'polynomial', 'degree': 3, 'coef0':...         0.337816
5  {'kernel': 'polynomial', 'degree': 3, 'coef0':...         0.317372
6  {'kernel': 'polynomial', 'degree': 2, 'coef0':...         0.344529
7  {'kernel': 'polynomial', 'degree': 2, 'coef0':...         0.346260
8  {'kernel': 'polynomial', 'degree': 4, 'coef0':...         1.852785
9  {'kernel': 'polynomial', 'degree': 2, 'coef0':...         0.344501
Score of best params using test data:
0.32711112055260577
```

Grid Search VS. Random Search...

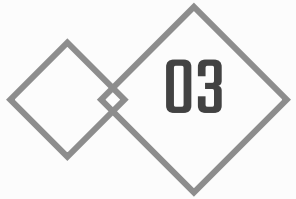


Hyperparameters Tuning

```
In [64]: #Kernal grid search
param_grid={'alpha':[0.1,0.3,0.5], 'kernel':["polynomial"], 'degree':[3],'coef0':[0.8,1,1.2]}
grid_search(KernelRidge()).grid_get(X_train,Y_train,X_test,Y_test,param_grid)
```

```
{'alpha': 0.5, 'coef0': 1.2, 'degree': 3, 'kernel': 'polynomial'} 0.3411977010560727
                                                params  mean_test_score
0  {'alpha': 0.1, 'coef0': 0.8, 'degree': 3, 'ker...         0.410609
1  {'alpha': 0.1, 'coef0': 1, 'degree': 3, 'kerne...         0.405574
2  {'alpha': 0.1, 'coef0': 1.2, 'degree': 3, 'ker...         0.402764
3  {'alpha': 0.3, 'coef0': 0.8, 'degree': 3, 'ker...         0.355958
4  {'alpha': 0.3, 'coef0': 1, 'degree': 3, 'kerne...         0.352429
5  {'alpha': 0.3, 'coef0': 1.2, 'degree': 3, 'ker...         0.350600
6  {'alpha': 0.5, 'coef0': 0.8, 'degree': 3, 'ker...         0.345666
7  {'alpha': 0.5, 'coef0': 1, 'degree': 3, 'kerne...         0.342781
8  {'alpha': 0.5, 'coef0': 1.2, 'degree': 3, 'ker...         0.341198
Score of best params using test data:
0.3286005136375086
```

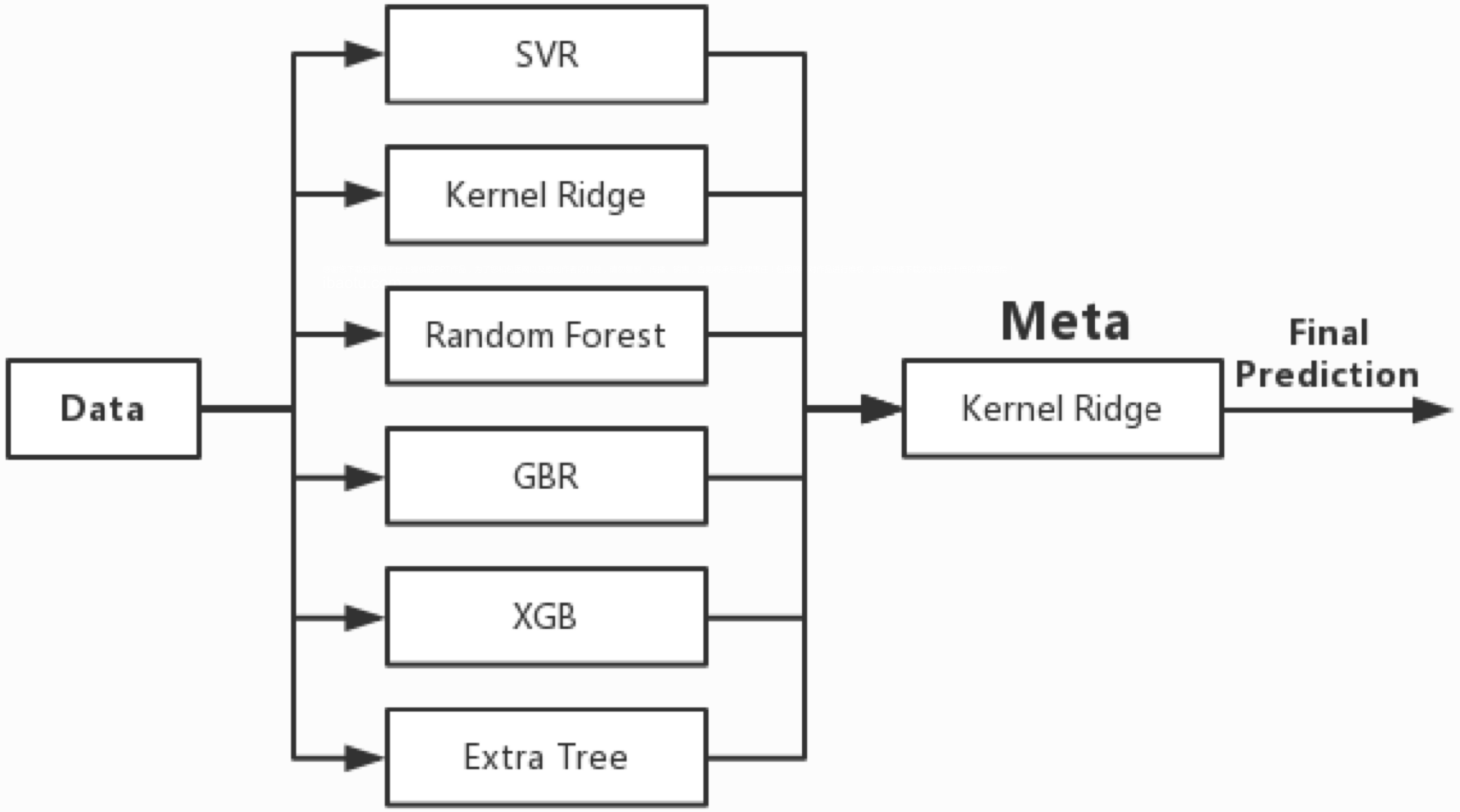
Very time-consuming



Ensemble Methods

Model	Training error	Testing error
Random Forest	0.2350	0.2317
Gradient Boost	0.2395	0.2354
SVR	0.2766	0.2614
Kernel Ridge	0.3174	0.3271
Extra	0.2448	0.2378
XGB	0.2225	0.2256
Average Six	0.2228	0.2214
Average Two (RF + XGB)	0.2222	0.2237
Stacking	0.2188	0.2201

03 Ensemble Methods



Result & Future Work

$$RMSE:$$

$$Baseline = 0.48$$

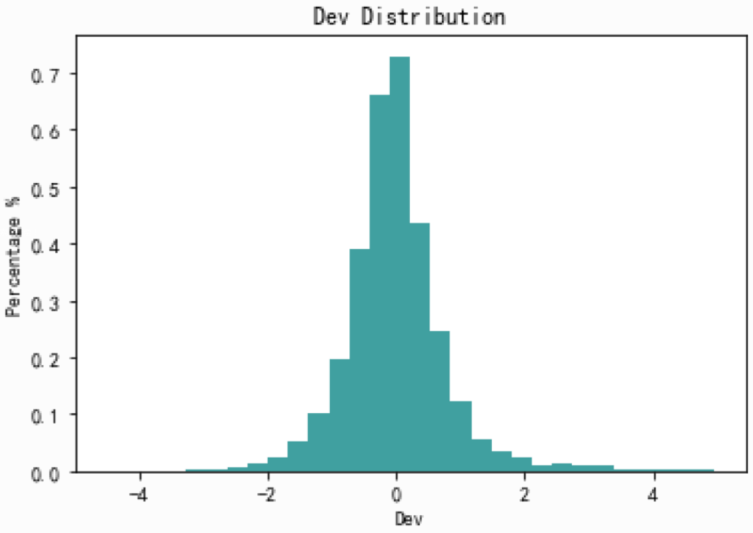
$$StackingModel = 0.22$$

$$RMSPE = \sqrt{\frac{1}{N}\sum_{i}^{N}\left(\frac{\hat{Y}_i - Y_i}{Y_i}\right)^2} = 0.26$$

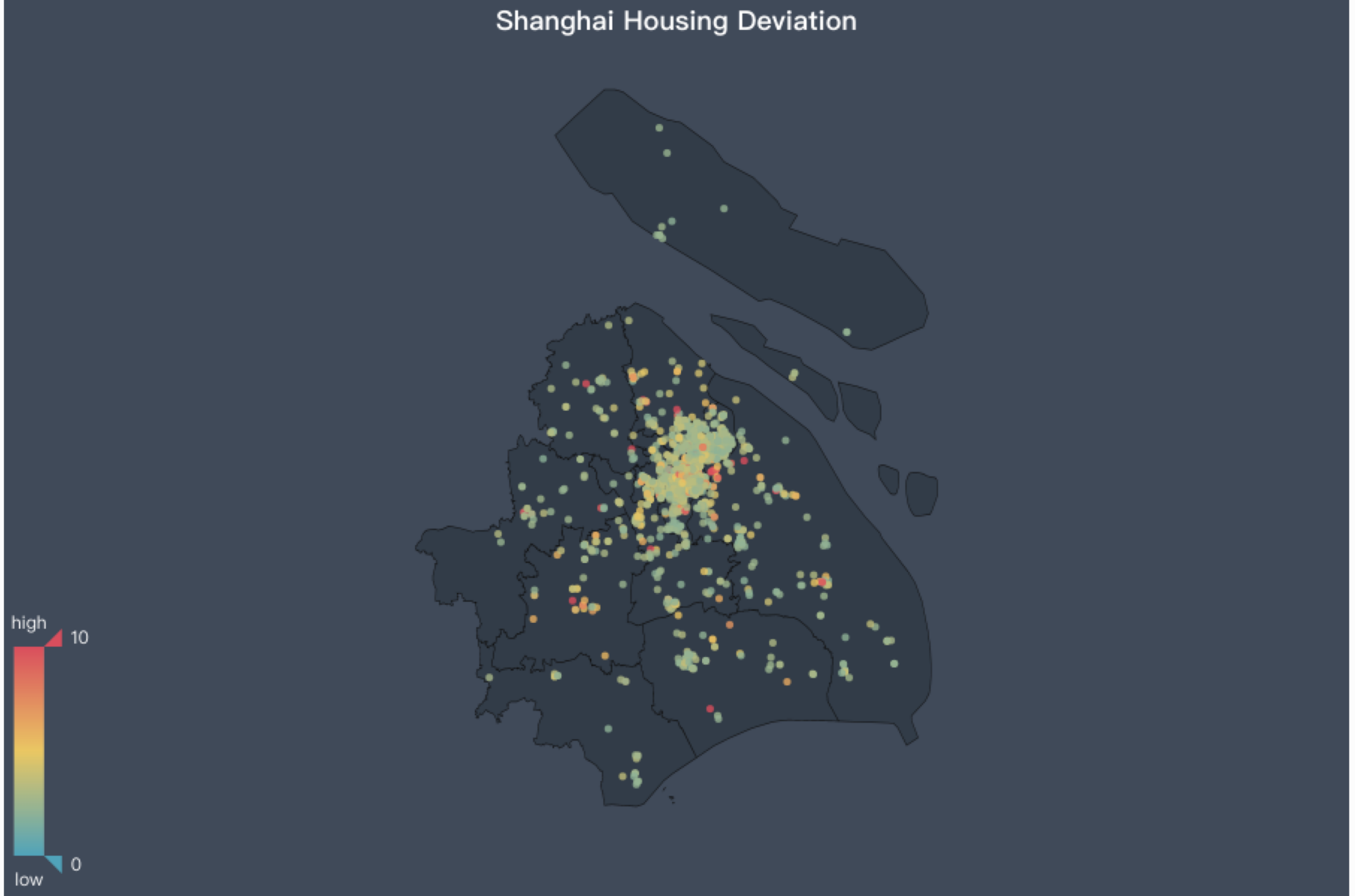
04 Result & Future Work

Deviation and Undervalued Units

$$De_i = \frac{\frac{\hat{Y}_i - Y_i}{Y_i}}{RMSPE}$$



Demo...


Shanghai Housing Deviation
high
10
0
low

04 Future Work

More features and data points;

Cross-sectional to Panel Data;

Parameter tuning for the stacking model;

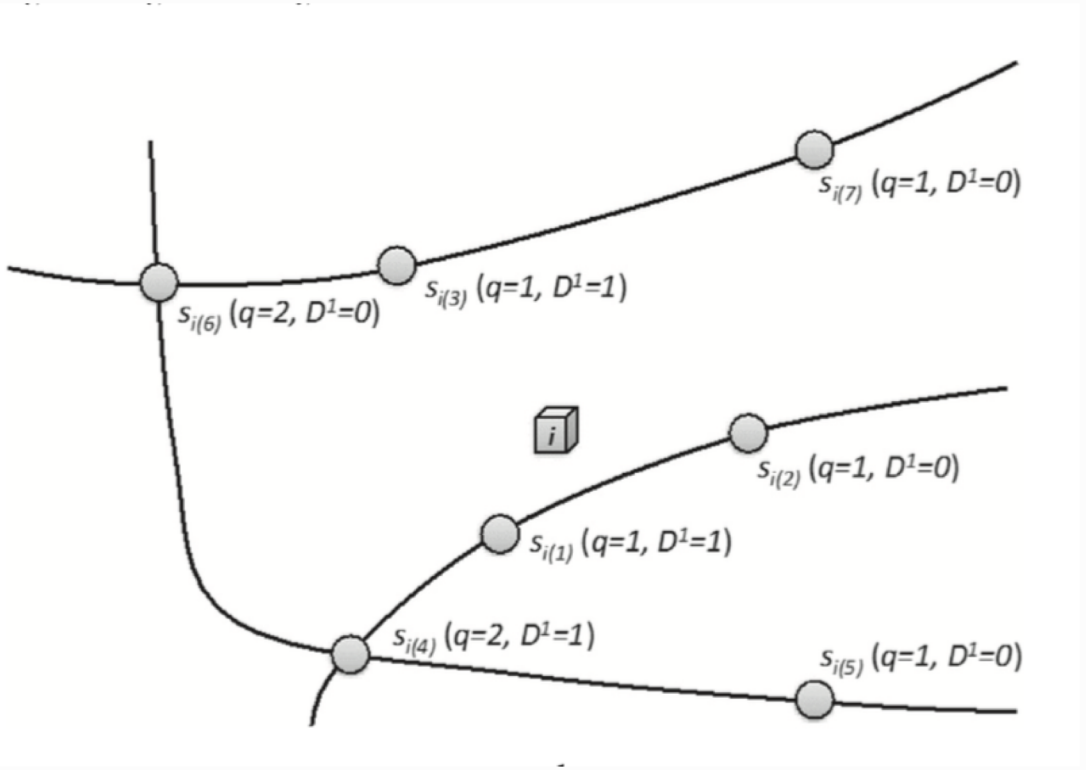
Do different cities, compare housing price with rental price……

04 Future Work

Currently only use Euclidean distance

Try well-developed accessibility measures

$$G_i^J = \sum_{j=1}^{J} \left(\sum_{k=1}^{K} D^k \left(s_{i(j)} \right) f^k \left(d_{i(j)}, q_{i(j)} \right) \right) + c_{(j)}.$$



Sadayuki, Taisuke. “Measuring the Spatial Effect of Multiple Sites: An Application to Housing Rent and Public Transportation in Tokyo, Japan.” *Regional Science and Urban Economics*, vol. 70, 2018, pp. 155–173., doi:10.1016/j.regsciurbeco.2018.03.002.

Thank you!