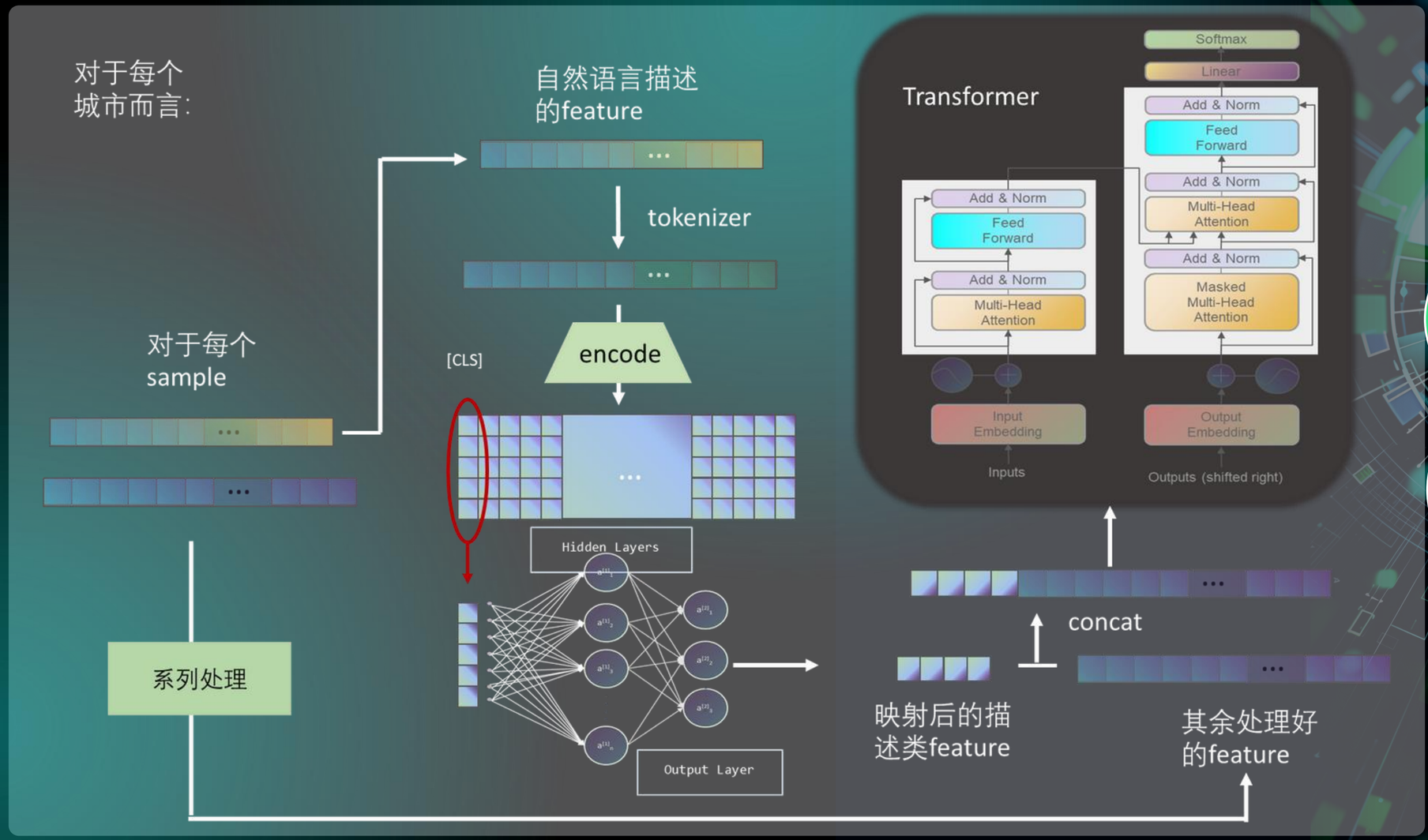
The era of artificial intelligence in finance has arrived！

基于transformer的深度学习模型预测房产价格



中国人民大学
RENMIN UNIVERSITY OF CHINA

QUANT
HACKATHON

段毅
2023200660

2025年6月12日

使用lazy_predict尝试所有回归模型

Model	# Adjus...	# R-Squ...	# RMSE	# Time ...	# m...
RandomForestRegressor	0.9693221716	0.9696576676	0.1531153080	698.74104690	0.0984852906
ExtraTreesRegressor	0.9683791809	0.9687249896	0.1554507649	2624.9218113	0.1031953345
BaggingRegressor	0.9650649111	0.9654469650	0.1633944241	60.861406803	0.1055787399
XGBRegressor	0.9595510262	0.9599933805	0.1758166740	12.174849510	0.1252961606
HistGradientBoostingRegressor	0.9420471237	0.9426809026	0.2104475159	9.2714400291	
LGBMRegressor	0.9419393974	0.9425743544	0.2106430213	4.1719877719	
DecisionTreeRegressor	0.9399224554	0.9405794698	0.2142705051	10.063408613	
ExtraTreeRegressor	0.9017920385	0.9028660513	0.2739552668	2.3603260517	
GradientBoostingRegressor	0.8774118055	0.8787524431	0.3060769539	116.63037919	
MLPRegressor	0.8571925026	0.8587542607	0.3303557044	34631.587220	
NuSVR	0.8398811290	0.8416322061	0.3498062508	30651.177115	
SVR	0.8397900914	0.8415421641	0.3499056801	3415.9610466	
AdaBoostRegressor	0.6431750199	0.6470772961	0.5221964695	55.675699472	
KNeighborsRegressor	0.6120791216	0.6163214658	0.5444749246	7.8357985019	
BayesianRidge	0.5900441455	0.5945274666	0.5597251903	19.067684650	
RidgeCV	0.5900441376	0.5945274588	0.5597251956	23.993312120	
Ridge	0.5900438269	0.5945271515	0.5597254077	1.2750403881	
TransformedTargetRegressor	0.5900437897	0.5945271147	0.5597254331	29.024758100	0.4242293493
LinearRegression	0.5900437897	0.5945271147	0.5597254331	22.388234853	0.4242293493
LassoLarsCV	0.5900283877	0.5945118811	0.5597359474	5.5095481872	0.4241522789
LassoLarsIC	0.5900267887	0.5945102996	0.5597370390	27.517242193	0.4241471165
LassoCV	0.5899324329	0.5944169757	0.5598014473	960.98595523	0.4240225333
ElasticNetCV	0.5899277575	0.5944123514	0.5598046387	794.31018328	0.4240265821
LarsCV	0.5888462775	0.5933426986	0.5605423366	6.1975870132	0.4243729511
Lars	0.5887052399	0.5932032034	0.5606384695	1.5165975093	0.4252349355

```python
from lazypredict.Supervised import LazyRegressor
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error
df = pd.read_pickle('data/df.pkl')
train_df = df[df['ID'] == -1].dropna()
pred_df = df[df['ID'] != -1]

X = train_df.drop(['价格', 'ID'], axis=1)
y = train_df['价格']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
# 拟合全部模型
reg = LazyRegressor(predictions=True, custom_metric=mean_absolute_error)
scores, predictions = reg.fit(X_train, X_test, y_train, y_test)
print(scores)
```

高维稀疏性

仍带来巨大挑战

核心处理

对7个城市分开建模

```
城市 0 的模型性能:
Ensemble Train - MAE: 0.0370, RMSE: 0.0501, Median AE: 0.0281
Ensemble Test - MAE: 0.0678, RMSE: 0.1000, Median AE: 0.0483

城市 1 的模型性能:
Ensemble Train - MAE: 0.0323, RMSE: 0.0441, Median AE: 0.0243
Ensemble Test - MAE: 0.1076, RMSE: 0.1510, Median AE: 0.0782

城市 2 的模型性能:
Ensemble Train - MAE: 0.0555, RMSE: 0.0726, Median AE: 0.0444
Ensemble Test - MAE: 0.0858, RMSE: 0.1186, Median AE: 0.0652

城市 3 的模型性能:
Ensemble Train - MAE: 0.0165, RMSE: 0.0222, Median AE: 0.0125
Ensemble Test - MAE: 0.0795, RMSE: 0.1174, Median AE: 0.0557

城市 4 的模型性能:
Ensemble Train - MAE: 0.0281, RMSE: 0.0367, Median AE: 0.0223
Ensemble Test - MAE: 0.0948, RMSE: 0.1354, Median AE: 0.0699

城市 5 的模型性能:
Ensemble Train - MAE: 0.0553, RMSE: 0.0760, Median AE: 0.0418
Ensemble Test - MAE: 0.1172, RMSE: 0.1605, Median AE: 0.0885

城市 6 的模型性能:
Ensemble Train - MAE: 0.0460, RMSE: 0.0623, Median AE: 0.0355
Ensemble Test - MAE: 0.0829, RMSE: 0.1147, Median AE: 0.0622

------------------------------------------------------------
Ensemble Model Performance:
Ensemble Train - MAE: 0.0829, RMSE: 0.1121, Median AE: 0.0638
Ensemble Test - MAE: 0.0972, RMSE: 0.1336, Median AE: 0.0734
```

一个有趣的现象

如果把不同的树模型的结果再求加权平均，结果会好于任何一个模型

自然语言的vectorization

采用BERT + Tokenizer + Encoder + MLP降维（到20维左右）

相比之下，TF-IDF生成的是非常高维（词汇表大小）且极度稀疏的向量

消融实验

添加该处理能使得模型从68.995提升到71.542

Final-Model的选择

基于Transformer架构的深度学习模型 (模型示意图见page1)

- 全局交互建模——Transformer
- 自动学习输入特征的高阶表示

核心组件

```python
class HousePriceModel(nn.Module):
    def __init__(self, bert_model, bert_dim=768,
                 hidden_dim=104, output_dim=1):
        super().__init__()
        self.bert_extractor = BERTFeatureExtractor(bert_model)
        self.bert_mapping = nn.Sequential(  # BERT特征映射层
            nn.Linear(bert_dim, hidden_dim),
            nn.ReLU(),
            nn.Linear(hidden_dim, 20)
        )

        self.transformer = nn.TransformerEncoder(  # Transformer层
            nn.TransformerEncoderLayer(
                d_model=104,  # 84(数值特征) + 20(BERT特征)
                nhead=4,
                dim_feedforward=256,
                dropout=0.1
            ),
            num_layers=2
        )

        self.output_layer = nn.Sequential(# 输出层
            nn.Linear(104, 50),
            nn.ReLU(),
            nn.Linear(50, output_dim)
        )
```

关键参数

NVIDIA GeForce RTX 3090

params：282,785

batch size = 4

learning rate=0.001

训练过程

```python
def train_model(model, train_loader, optimizer,
                criterion, device, avg_price):
    model.train()
    total_loss = 0

    for batch in train_loader:
        numeric_features = batch['numeric_features'].to(device)
        input_ids = batch['input_ids'].to(device)
        attention_mask = batch['attention_mask'].to(device)
        # 计算价格与平均价格的差值
        targets = batch['price'].to(device) - avg_price

        optimizer.zero_grad()
        outputs = model(numeric_features, input_ids, attention_mask)
        loss = criterion(outputs.squeeze(), targets)

        loss.backward()
        optimizer.step()

        total_loss += loss.item()

    return total_loss / len(train_loader)
```

最终结果

Metrics	MAE	RMSE	Median-AE
In-sample	0.0908	0.1282	0.0669
Out-of-sample	0.0387	0.0520	0.0298
Cross-validation	0.0647	0.0713	/
Total N	84127		
Datahub Score	82.984	rank	12

01 模型性能表

Model Performance Table

02 特征重要性

feature importance table

	Feature	Importance
1	建筑面积	0.6584625375765866
11	租金	0.05966229494938912
12	室	0.031696273857496654
16	总楼层数	0.026730065848364248
176	lat2central	0.023055396188334985
68	房屋用途_车库	0.021293323764995643
6	停车位	0.019971898980227824
177	lon2central	0.017156862397494047
15	卫	0.01403575956969068
4	供热费	0.012255279101258783
34	建筑结构_钢混结构	0.00795446101378409
9	绿 化 率	0.007275323860694387
35	lift_ratio	0.005896592715914802
8	容 积 率	0.005711120774869705
10	物 业 费	0.004498740878048506
7	停车费用	0.0037275624945605507
17	楼层位置encode_-1	0.003617415994976884
75	房本满五年	0.0013158757995790657
5	燃气费	0.0012431929731892184
57	房屋用途_别墅	0.0011560245907481022
19	楼层位置encode_1	0.001059738977756138
24	south	0.0008610510905891637
28	装修情况_简装	0.0008197151675851962
21	楼层位置encode_3	0.0007738685090922682