




中國人民大學
RENMIN UNIVERSITY OF CHINA

人工智能与Python程序设计 期中展示

 Team3: 梁东成 张凌蔚 靳沛晓

01

数据处理:

clean_complex_features_optimized处理各种格式（单位、范围、楼层、户型）。

性能优化:

通过向量化 (.str.extractall) 和调整 GridSearchCV 流程（先预处理）来大幅缩短运行时间。

指标计算的准确性:

calculate_mae_rmse_original能够确保最终报告的MAE在原始价格尺度上计算，能处理各种数值稳定性问题。

模型比较:

展示最终的性能表。

Price

Rent

clean_unit: 处理带单位的列；向量化.str.extractall和groupby().mean(): 从文本（物业费, 燃气费,.....）中提取数值并计算均值，生成“_均值”列

数据预处理: **numeric_transformer Pipeline**（数值）**categorical_transformer Pipeline**（文本）

“SimpleImputer(strategy='median')”: 中位数填充缺失值

“FunctionTransformer(np.log1p)”: 对数变换 ($\log(1+x)$)，处理数据偏度。

“SimpleImputer(strategy='median')”: 再次中位数填充（处理log变换后可能产生的NaN/Inf）。

StandardScaler(): 标准化（数据缩放到均值为0，方差为1）。

数值特征: 所有数值特征对数变换np.log1p; **多项式**, 创建'容积率_sq'; **交互项**: '套内面积_比'（套内面积/建筑面积）; **Dummies**: OneHotEncoder 为所有分类特征创建了虚拟变量。

建模: param_grids 定义了要搜索的超参数: alpha (控制 L1/L2 正则化强度) 和 l1_ratio (ElasticNet 中 L1/L2 的比例); 使用了 GridSearchCV 来自动寻找 param_grids 中定义的最佳超参数组合。

6 folds cross validation: kf = KFold(n_splits=6) 定义 6 折。GridSearchCV (Lasso, Ridge, ElasticNet) 和手动 CV (OLS) 都基于这个 kf 计算了交叉验证 MAE (cv_mae)。 **In-sample**: 在 X_train_transformed 上预测并计算 mae_train; **Out-of-sample**: 在 X_valid_transformed 上预测并计算 mae_valid; **All models + best**: result_df 包含了所有四个模型的指标，代码会自动找出 CV MAE 最低的模型，并将其在 result_df 中的名字改为 "Best Linear Model"。

计算结果: calculate_mae_rmse_original 函数内部使用 np.expm1() 将对数尺度的预测值和真实值转换回原始价格（或租金）尺度，然后才调用 mean_absolute_error 计算 MAE。确保所有报告的 MAE 在原始尺度上。

以Price为例: 最佳是Ridge。交叉验证 MAE 是衡量模型泛化能力（在未知数据上表现）最重要的指标，因此选择 Ridge 是合理的；几乎没有过拟合（和CV相似）；相对误差19.3%到19.7%

Metrics	In-sample MAE	In-sample RMAE	Out-of-sample MAE	Out-of-sample RMAE	Cross-validation MAE	Cross-validation RMAE
OLS	410056.4941	0.192760158	405740.841	0.190731448	410888.1108	0.193151086
LASSO	417956.9578	0.196474024	414388.5031	0.194796558	418347.8294	0.196657766
Best Linear Model	410019.4825	0.19274276	405728.6176	0.190725702	410824.1498	0.19312102
ElasticNet	415565.3534	0.195349774	411454.0388	0.193417119	416135.2793	0.195617686

Metrics	In-sample MAE	In-sample RMAE	Out-of-sample MAE	Out-of-sample RMAE	Cross-validation MAE	Cross-validation RMAE
Best Linear Model	104514.5577	0.188770596	104586.5771	0.188900675	104894.0205	0.189455969
LASSO	105818.9167	0.191126485	106098.544	0.191631538	106126.3928	0.191681838
Ridge	104717.1325	0.18913648	104800.3341	0.189286756	105069.1578	0.189772296
ElasticNet	105343.8308	0.190268401	105515.6509	0.190578737	105683.1795	0.190881321

team 3 张凌蔚 2022202717

Metrics	In sample	out of sample	Cross-validation	Kaggle Score
OLS	0.76	0.77	0.76	-36.79
LASSO	0.76	0.76	0.76	-22.17
Best Linear Model	0.76	0.77	0.76	-36.79

交易时间：获取交易年份、上次交易
时间间隔、是否首次交易

```
def process_trade_dates(df):  
    # 统一时间格式  
    df['交易时间'] = pd.to_datetime(df['交易时间'], errors='coerce')  
    df['上次交易'] = pd.to_datetime(df['上次交易'], errors='coerce')  
    df['交易年份'] = df['交易时间'].dt.year  
  
    # 新增首次交易标记  
    df['首次交易'] = df['上次交易'].isna().astype(int)  
  
    # 计算交易间隔天数  
    df['交易间隔天数'] = (df['交易时间'] - df['上次交易']).dt.days  
  
    # 首次交易的交易间隔填0  
    df.loc[df['首次交易'] == 1, '交易间隔天数'] = 0  
  
    return df
```

楼层处理逻辑：根据语言描述生成相对高度 - 用相对高度乘
总楼层估计绝对楼层 - 单独区分别墅

```
# 所在楼层  
def process_floor_vectorized(df, villa_col='用途_别墅', region_col='板块', floor_col='所在楼层'):  
    """  
    处理楼层列，生成三列：  
    rel_floor: 相对高度 (0~1, 底层0.2, 顶层0.9)，地下室为0  
    total_floor: 总楼层数  
    est_floor: 估计的该房源楼层数 (绝对层数)，地下室为-1  
    别墅直接填为0  
    """  
    rel_map = {'地下室': 0.0, '底': 0.1, '低': 0.3, '中': 0.5, '高': 0.7, '顶': 0.9}  
  
    # 提取总楼层数  
    total_floors = df[floor_col].str.extract(r'共(\d+)层')[0].astype(float)  
  
    # 提取相对楼层  
    def map_rel(text):  
        for k, v in rel_map.items():  
            if k in str(text):  
                return v  
        return 0.5 # 默认中层  
    rel_floor = df[floor_col].map(map_rel)  
  
    # 计算绝对楼层  
    est_floor = np.round(rel_floor * total_floors).fillna(1).astype(float)  
  
    df['rel_floor'] = rel_floor  
    df['total_floor'] = total_floors  
    df['est_floor'] = est_floor  
  
    # 地下室绝对楼层为-1  
    est_floor[df[floor_col].str.contains('地下室', na=False)] = -1  
  
    # 别墅全部填为0  
    villa_idx = df[df[villa_col]==1].index  
    df.loc[villa_idx, ['rel_floor', 'est_floor', 'total_floor']] = 0  
  
    return df
```

靳沛峒 2022202683

===== PRICE Metrics (MAE) =====

价格 模型性能汇总:

	Metrics	In sample	In sample RMAE	Out of sample	Out of sample RMAE	Cross-validation	CV RMAE
	LASSO	264368.093750	0.154154	487137.596984	0.218064	265271.728779	0.154681
	Elastic Net	267149.100409	0.155776	482963.737019	0.216196	268083.842920	0.156321
	Ridge	304332.351388	0.177458	488157.436197	0.218521	300646.354675	0.175308
	OLS	336124.716858	0.195996	513668.954146	0.229941	341350.976229	0.199043
Best Linear Model		264368.093750	0.154154	487137.596984	0.218064	265271.728779	0.154681

===== RENT Metrics (MAE) =====

租金 模型性能汇总:

	Metrics	In sample	In sample RMAE	Out of sample	Out of sample RMAE	Cross-validation	CV RMAE
	Elastic Net	68904.613870	0.145718	111989.082549	0.192458	68999.762421	0.145919
	LASSO	69038.799461	0.146002	113373.982993	0.194838	69173.617374	0.146287
	Ridge	74272.195431	0.157069	112957.155880	0.194122	72987.944734	0.154353
	OLS	80876.692606	0.171036	119694.787245	0.205700	80570.832140	0.170390
Best Linear Model		68904.613870	0.145718	111989.082549	0.192458	68999.762421	0.145919

```
FAST_DEMO      = True
MAX_CV_ROWS     = 30000
CV_FOLDS        = 6
RANDOM_STATE     = 111
```

FAST_DEMO 模式下限制参与CV的样本规模

```
cv = KFold(n_splits=CV_FOLDS, shuffle=True, random_state=RANDOM_STATE)
X_cv, y_cv_raw = X_tr, y_tr_raw
if FAST_DEMO and X_tr.shape[0] > MAX_CV_ROWS:
    rng = np.random.RandomState(RANDOM_STATE)
    idx = rng.choice(np.arange(X_tr.shape[0]), size=MAX_CV_ROWS, replace=False)
    X_cv, y_cv_raw = X_tr.iloc[idx], y_tr_raw.iloc[idx]
print(f"[{dataset_name}] CV样本: {X_cv.shape[0]} / {X_tr.shape[0]} (FAST_
```

```
lasso_grid = {
    "regressor__reg__alpha": [
        np.r_[np.logspace(-4, -1, 5), np.logspace(-1, 2, 8)] if FAST_DEMO
        else np.logspace(-4, 2, 20)
    ]
}
```

---- OLS ----

print(f"[{dataset_name}] 训练 OLS")

ols_pipe = Pipeline([("pre", preproc), ("reg", LinearRegression())])

ols_ttr = TransformedTargetRegressor(regressor=ols_pipe, func=np.log1p, inverse_func=np.expm1)


cv_mae_ols = -np.mean(cross_val_score(ols_ttr, X_cv, y_cv_raw, cv=cv, scoring="neg_mean_absolute_error", n_jobs=-1))

ols_ttr.fit(X_tr, y_tr_raw)



中國人民大學
RENMIN UNIVERSITY OF CHINA

谢谢垂听!

 Team3: 梁东成 张凌蔚 靳沛晓