Peter the Great St. Petersburg Polytechnic University
Institute of Computer Science and Technology
Graduate School of Intelligent Systems and Supercomputing Technologies

Course work report
Determination of real estate prices

In the discipline "seminar on specialty"

Completed by students
of group 5130203/20102

I. N. Duro
I. V. Zhdanova
D. A. Shulzhik
D. V. Sereda
A. R. Karakeschishyan

Teacher

Espinola Rivera Holger Elias

Saint Petersburg
2024

1. INTRODUCTION

Determining the market value of residential real estate is an urgent task that is of great importance for both individuals and businesses. Buying or selling a home is an important step that requires careful analysis and informed decision–making.

Traditional methods of valuing real estate can be subjective, labour-intensive and inaccurate. In a rapidly developing real estate market and availability of large volume of data on properties for sale, the evaluation process needs to be automated.

Thus, choosing apartments as the object of study allows us to focus on the key factors that determine the price of housing and develop an effective machine learning model for predicting market value that can be applied in practice.

Tasks:

1. Select data set.
2. Pre-process the data, eliminating gaps and outliers.
3. Build and train several models to predict prices.
4. Evaluate the quality of the models and choose the most suitable one.

2. SOLUTION METHOD

2.1. Selected data set

The Russia Real Estate 2018-2021 dataset was selected for the study because it:

- Contains up-to-date data: covers the period from 2018 to 2021 and provides the most recent data among available datasets.
- Has a wide coverage: includes information on real estate objects all over Russia for three years, which allows us to analyze both the geographical distribution of prices and their dynamics over time.
- Contains a fairly large amount of data: more than 540 thousand objects.
- Includes a variety of features that allow us to take into account various factors affecting the price of apartments.

2.2. Statistical analysis

- A matrix of feature correlations has been constructed.
- Important and uninformative features have been identified. Despite the theoretical weakness of some features ("month", "building_type", "object_type"), their use has reduced the model error.

In raw form, the data takes up 400 MB, in processed form and in another format - ~60-70 MB. Apache Parquet is used as a new format due to its storage efficiency

2.3. Models

1. **RandomForest.**

Random forest is an ensemble method based on constructing a set of decision trees and averaging their predictions. It uses the bagging method (bootstrap aggregating).

RandomForest model was trained on CPU (i7-12650h)

Specifications:

RandomForestRegressor - 40 trees, all features are used.

Full config -{'bootstrap': True, 'ccp_alpha': 0.0, 'criterion': 'friedman_mse', 'max_depth': None, 'max_features': 1.0, 'max_leaf_nodes': None, 'max_samples': None, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 1, 'min_samples_split': 2, 'min_weight_fraction_leaf': 0.0, 'monotonic_cst': None, 'n_estimators': 40, 'n_jobs': -1, 'oob_score': False, 'random_state': 42, 'verbose': 0, 'warm_start': False}

2. **CatBoost**.

The core mechanism of CatBoost is gradient boosting on decision trees. Gradient boosting builds a sequential model by adding new models that try to correct the mistakes of the previous ones. The overall model is represented as a sum of individual models.

The model was trained on a GPU (RTX 3050 Ti Mobile).

Specifications:

CatBoostRegressor - CatBoostRegressor(iterations=400,

```
    learning_rate=0.5,
    depth=10,
    l2_leaf_reg=3,
    loss_function="RMSE",
    eval_metric="MAE",
    early_stopping_rounds=100,
    random_seed=42,
    verbose=100,
    task_type="GPU",
)
```

3. **Self-written neural network.**

The model was trained on a GPU (RTX 3050 Ti Mobile).

Specifications:

3 hidden layers - 128, 64, 32 neurons. Experiments have shown that this is the optimal model among those that were trained, batch size is 2048.

```
model = tf.keras.Sequential(
[
tf.keras.layers.Dense(128, activation="relu", input_shape=(X_train.shape[1],)),
tf.keras.layers.Dense(64, activation="relu"),
tf.keras.layers.Dense(32, activation="relu"),
tf.keras.layers.Dense(1), # Linear activation for regression
]
)
model.compile(optimizer="adam", loss="mse", metrics=["mae"])
```

3. RESULTS AND BENCHMARKS

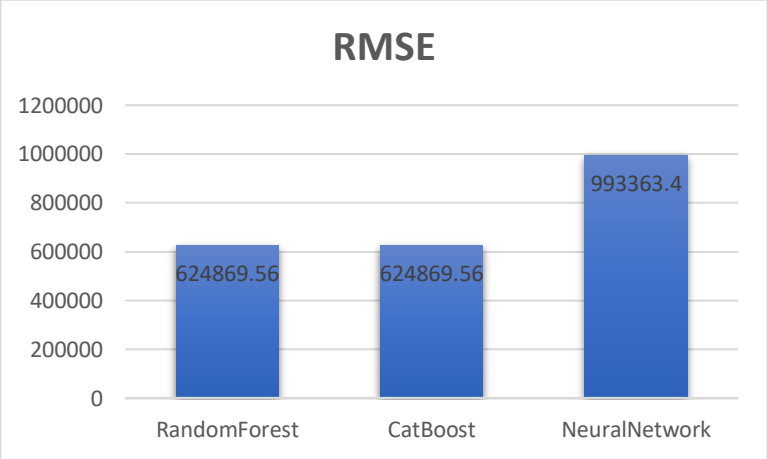

Figure 1 - RMSE

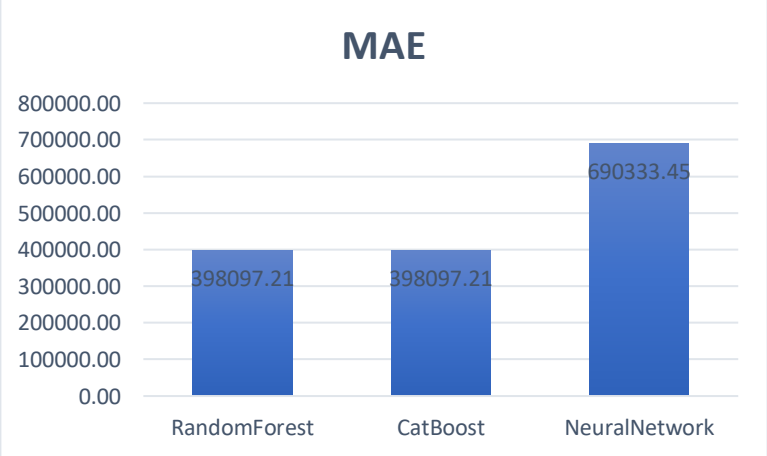

Figure 2 -MAE

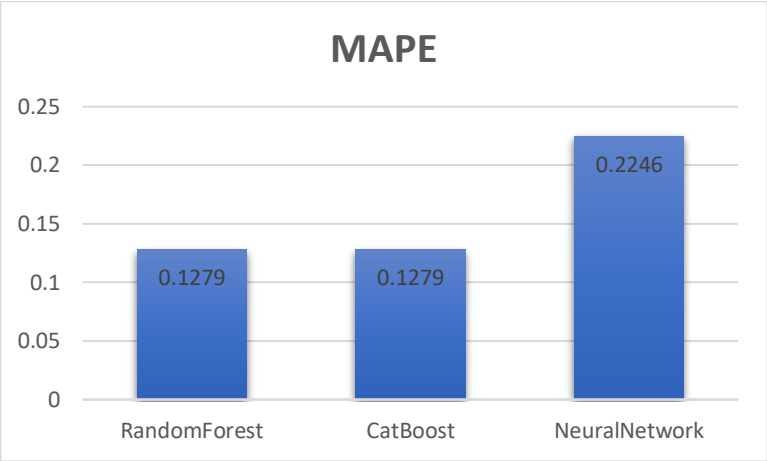

Figure 3 - MAPE

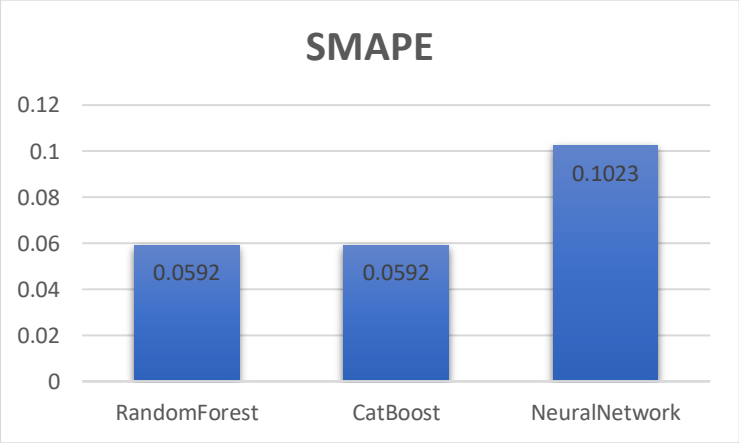

Figure 4 - SMAPE

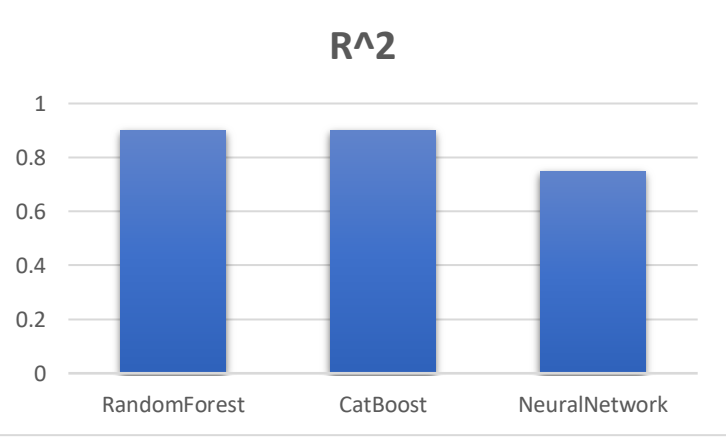

Figure 5 – R^2

1. Random Forest

RMSE: 624,869.56

This value indicates the standard deviation of the residuals (prediction errors). A lower RMSE means better model accuracy. Here it is relatively high, suggesting that predictions are spread out quite a bit around the actual values.

MAE: 398,097.21

The MAE tells us the average magnitude of errors in the predictions. It is in the same scale as the target variable (price), and here it suggests that on average, predictions deviate by around 398k.

MAPE: 0.1279 (12.79%)

MAPE shows the prediction error as a percentage of the actual values. This value of 12.79% means the model's predictions are, on average, off by about 13%.

SMAPE: 0.0592 (5.92%)

The SMAPE, which handles both over- and under-predictions more symmetrically, shows an error of about 5.92%, indicating a relatively smaller deviation when measured against both the prediction and actual values.

(R^2): 0.9007

The (R^2) value is quite high, indicating that 90.07% of the variance in the target variable (price) is explained by the model. This suggests a very good fit.

2. CatBoost

RMSE: 624,869.56

Identical to the Random Forest, which indicates the CatBoost model has a similar level of error in its predictions.

MAE: 398,097.21

Same as Random Forest, indicating the average error in predictions is similar for both models.

MAPE: 0.1279 (12.79%). Again, this is identical to Random Forest, suggesting that the percentage deviation in the predicted values is the same.

SMAPE: 0.0592 (5.92%). Same SMAPE value as Random Forest, confirming that both models have a similar performance in terms of absolute percentage errors.

(R^2): 0.9007. Identical (R^2) value to Random Forest, meaning that both models explain the same proportion of variance in the data.

3. Self-written neural network

The RMSE is much higher than both Random Forest and CatBoost, indicating that the Neural Network model's predictions are spread further from the actual values, implying poorer accuracy.

The MAE is also higher, suggesting that, on average, the Neural Network's predictions deviate more from the actual values compared to the other two models.

The MAPE is significantly higher than both Random Forest and CatBoost. It suggests that the Neural Network's predictions are off by about 22.5% on average, which is quite a large error percentage.

The SMAPE is also higher compared to the other two models, indicating higher relative errors when considering both over- and under-predictions.

The (R^2) value is much lower than that of Random Forest and CatBoost. It indicates that the Neural Network explains only 74.91% of the variance in the data, which is lower than both Random Forest (90.07%) and CatBoost (90.07%).

Random Forest and CatBoost are the best models. They have identical performance across all metrics, with high (R^2) values (90.07%) and relatively low errors in RMSE, MAE, MAPE, and SMAPE.

Neural Network is the worst performer. While it still performs reasonably well, its (R^2) (0.7491) is significantly lower, indicating poorer model fit. The higher RMSE, MAE, MAPE, and SMAPE suggest that the Neural Network's predictions are less accurate compared to both Random Forest and CatBoost.

CONCLUSION

The best model are the CatBoost and Random Forest models, both of which performed identically across all evaluation metrics. These models provided the best combination of accuracy and robustness, as evidenced by their high (R^2) values of 0.9007. This indicates that the models were able to explain approximately 90% of the variance in the target variable (price), suggesting a strong fit to the data. Additionally, their RMSE and MAE values were relatively low, indicating that the errors in their predictions were minimal.

CatBoost is known for efficiently handling categorical features and large datasets. It uses gradient boosting, which is particularly effective in capturing complex relationships between features.

Both Random Forest and CatBoost are robust to overfitting due to their ensemble nature, where multiple models (trees) are combined to make the final prediction. This leads to more stable and generalizable results.

Limitations and Accuracy

While both CatBoost and Random Forest performed well, their accuracy will never be 100%, as no model can perfectly predict every data point, especially in real-world scenarios. For instance, an R^2 value of 0.9007 means that 9.93% of the variance in the price remains unexplained, which implies that there are still factors affecting the property prices that these models couldn't capture.

Accuracy will vary across different datasets. This dataset is specific to real estate in Russia, and if the model were applied to a different market or dataset, its performance might change. Moreover, feature selection and quality of data play a crucial role in model performance. Missing

or incorrect data can lead to degraded performance, and more sophisticated data preprocessing might be required to improve the model's generalization ability.

The Neural Network model, while still capable of making predictions, performed significantly worse than both CatBoost and Random Forest. Its (R^2) value of 0.7491 indicates that it explained only 74.91% of the variance in the target variable. The neural network's performance could be affected by several factors:

- Data Complexity: it often requires large amounts of data to train effectively.
- Overfitting: Neural networks are more prone to overfitting, especially if the dataset is not sufficiently large or diverse, which could lead to worse generalization to unseen data.
- Optimization Issues: Neural networks also depend heavily on the choice of architecture, learning rate, and other hyperparameters.
- Gradient Attacks

One significant limitation of neural networks is their susceptibility to gradient-based attacks or adversarial examples. This issue arises because neural networks rely on gradient descent methods to optimize the model during training. In the presence of small, carefully crafted perturbations in the input data, the network's weights may shift dramatically, leading to inaccurate predictions or misclassifications.

During training, the neural network's weights are adjusted based on gradients of the loss function. However, when small perturbations (or "attacks") are introduced in the data, the network can "overfit" to these small changes, leading to incorrect predictions.

In contrast, Random Forest and CatBoost are less sensitive to such attacks because:

- Both models aggregate the predictions of multiple trees or weak learners. This ensemble nature makes them more robust to small changes in individual predictions.
- These models do not rely on gradient-based optimization, which makes them less susceptible to the types of small input changes that can confuse neural networks.