Предсказание цены на аренду жилья в Нью-Йорке в 2019 году

Мною был произведен анализ данных а так же кодирование категориальных признаков

- Основные методы кодирования строковых данных
- 1) **Label Encoding то есть, каждому признаку соответствует своя уникальная метка. Например** `Bronx -> 0, Brooklyn -> 1`
- 2) **One-Hot Encoding то есть, каждый признак кодируется двоичным ключом. Например** `Bronx -> 1 0 0, Brooklyn - > 0 1 0 Manhattan -> 0 0 1`
- **3) Ordinal Encoding то есть, кодирование по порядку(когда нам важен порядок). Например школа -> 0, бакалавр -> 1, магистратура -> 2 или** XS –> 0, S -> 1, M -> 2, L -> 3, XL -> 4
- **4) Embedding то есть, каждая категория отображается в уникальный вектор и мы хотим чтобы близкие по смыслу категории были в этом векторном пространстве тоже близки.**

Метрики используемые для оуенки качества моделей

- 1) RMSE (Root Mean Squared Error)
- 2) MAE (Mean Absolute Error)
- 3) R^2 Score (R-squared)

1) RMSE (Root Mean Squared Error)

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

RMSE (корень из среднеквадратичной ошибки) — это метрика, которая показывает, насколько сильно предсказания модели отличаются от истинных значений. Она измеряется в тех же единицах, что и целевая переменная (например, если целевая переменная — цена в долларах, то RMSE тоже будет в долларах).

Чем меньше RMSE, тем лучше модель.
RMSE чувствителен к выбросам, так как ошибки возводятся в квадрат (большие ошибки влияют сильнее).

2) MAE (Mean Absolute Error)

$$\mathrm{MAE} = \frac{1}{N}\sum |y_i - \hat{y}_i|$$

- MAE (средняя абсолютная ошибка) — это метрика, которая показывает среднюю абсолютную разницу между истинными и предсказанными значениями. Как интерпретировать?
- Чем меньше MAE, тем лучше модель. MAE менее чувствителен к выбросам, чем RMSE, потому что ошибки не возводятся в квадрат.

Сравнение RMSE с MAE

- **RMSE** (квадратичная ошибка):
  — Сильно штрафует **большие ошибки** (т.к. ошибки возводятся в квадрат).
- **MAE** (линейная ошибка):
  — Штрафует все ошибки **пропорционально их величине**.
  — Менее чувствителен к выбросам.
- NB: Если в данных есть редкие, но очень большие ошибки (например, аномальные значения цены дома), RMSE покажет их влияние, а MAE — нет.

3) R^2 Score (R-squared)

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

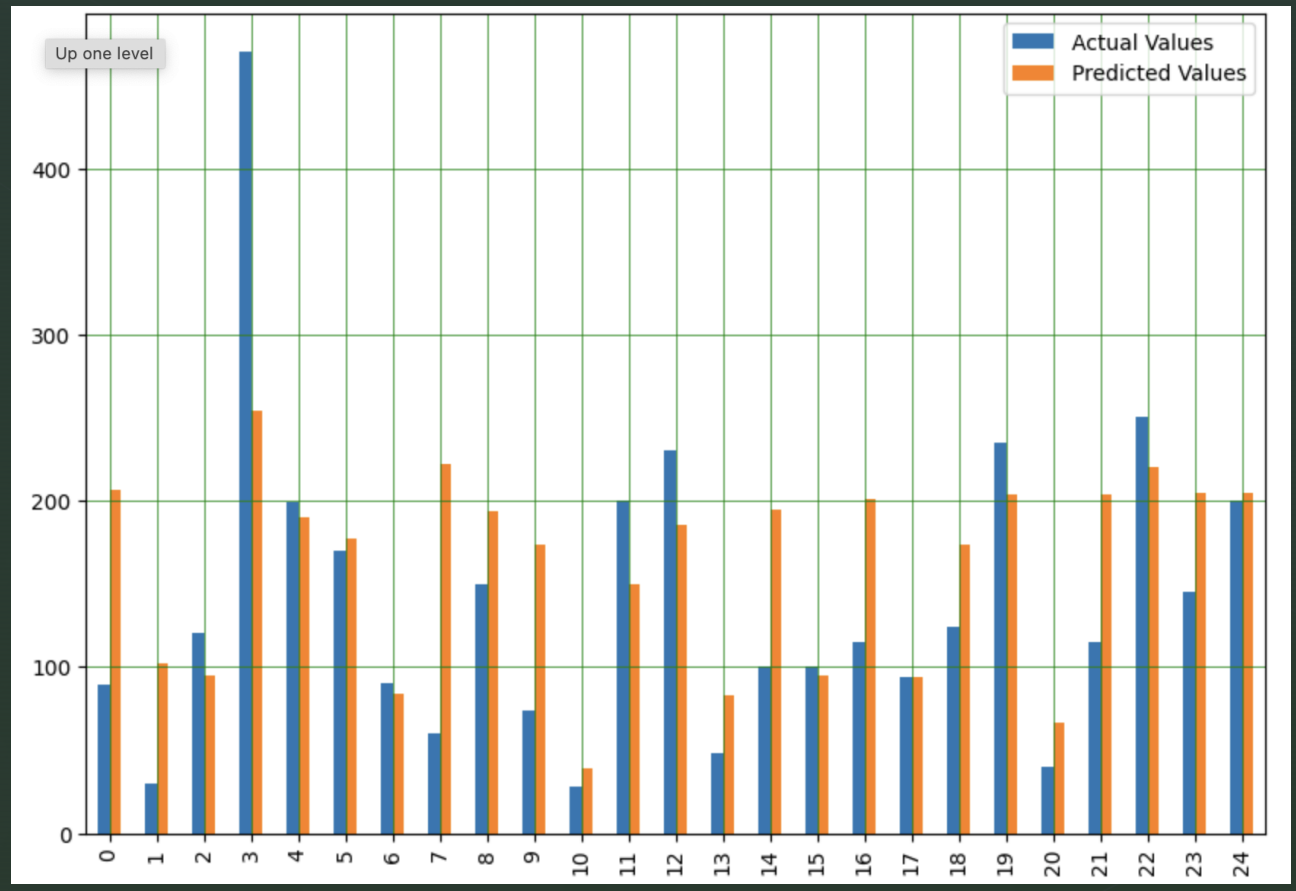
- R^2 Score (коэффициент детерминации) — это метрика, которая показывает, насколько хорошо модель объясняет дисперсию (разброс) целевой переменной. Она принимает значения от 0 до 1.
- R^2 =1 (или 100%): Модель идеально объясняет все изменения в данных. R^2 =0 (или 0%): Модель не объясняет никаких изменений в данных (предсказания равны среднему значению). Отрицательные значения R^2 возможны, если модель работает хуже, чем просто предсказание среднего значения. Например если R^2 =9.21%. Это означает, что модель объясняет только 9.21% дисперсии целевой переменной.

Используемые методы машинного обучения

- 1) линейная регрессия
- 2) случайный лес
- 3) градиентный бустинг

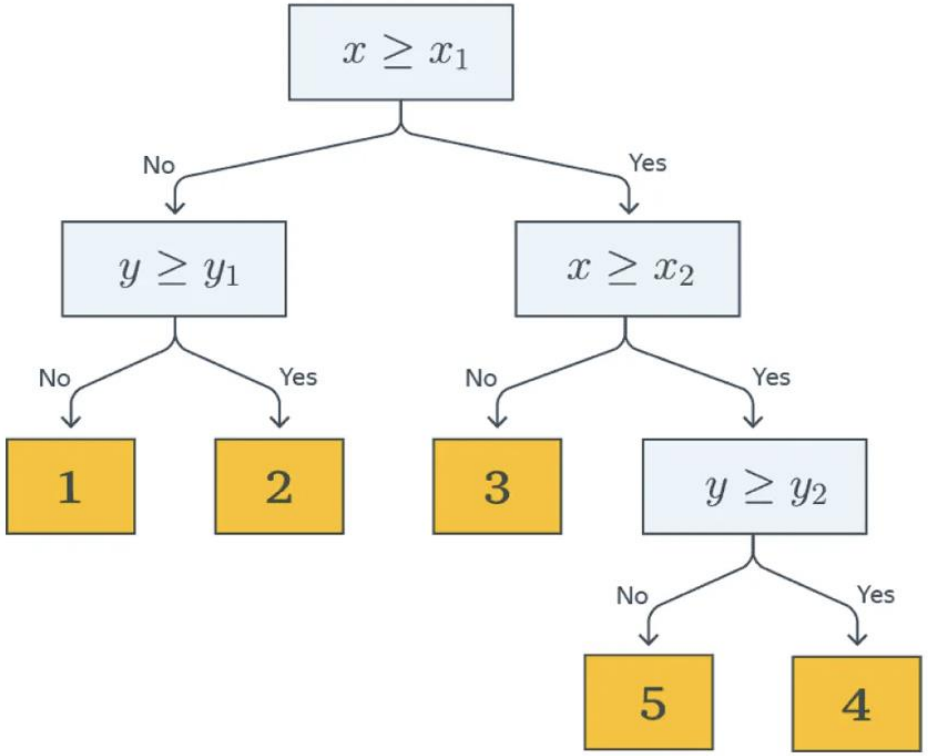
1) линейная регрессия

- Результат по метрикам: `RMSE: 200.40 MAE: 74.11 R2 Score: 9.21%`



2) случайный лес

- Начнем с понятия случайное дерево. Предположим стоит задача классификации на 5 классов. Вот как выглядит дерево решений:



Каждое дерево разбивает задачу на простые решающие правила.

Объекты имеют два признака. Решение о том, к какому классу будет отнесён текущий объект выборки, будет приниматься с помощью прохода от корня дерева к некоторому листу. То есть последовательно сравниваем элемент с предикатами. Как только мы дошли до листа, мы присваиваем объекту ответ, записанный в вершине.

Ансамбль алгоритмов из решающих деревьев

- Из исходных данных N раз случайно выбираются подвыборки с повторением (обычно того же размера, что и исходные данные).
- Для каждой подвыборки строится отдельное решающее дерево.
- Каждое дерево обучается независимо от других.
- Каждое дерево "голосует" за класс, итоговый ответ — мажоритарный выбор (например, если 80 деревьев из 100 проголосовали за класс "1", это и будет предсказанием).

Как подбирать гиперпараметры в случайном лесе

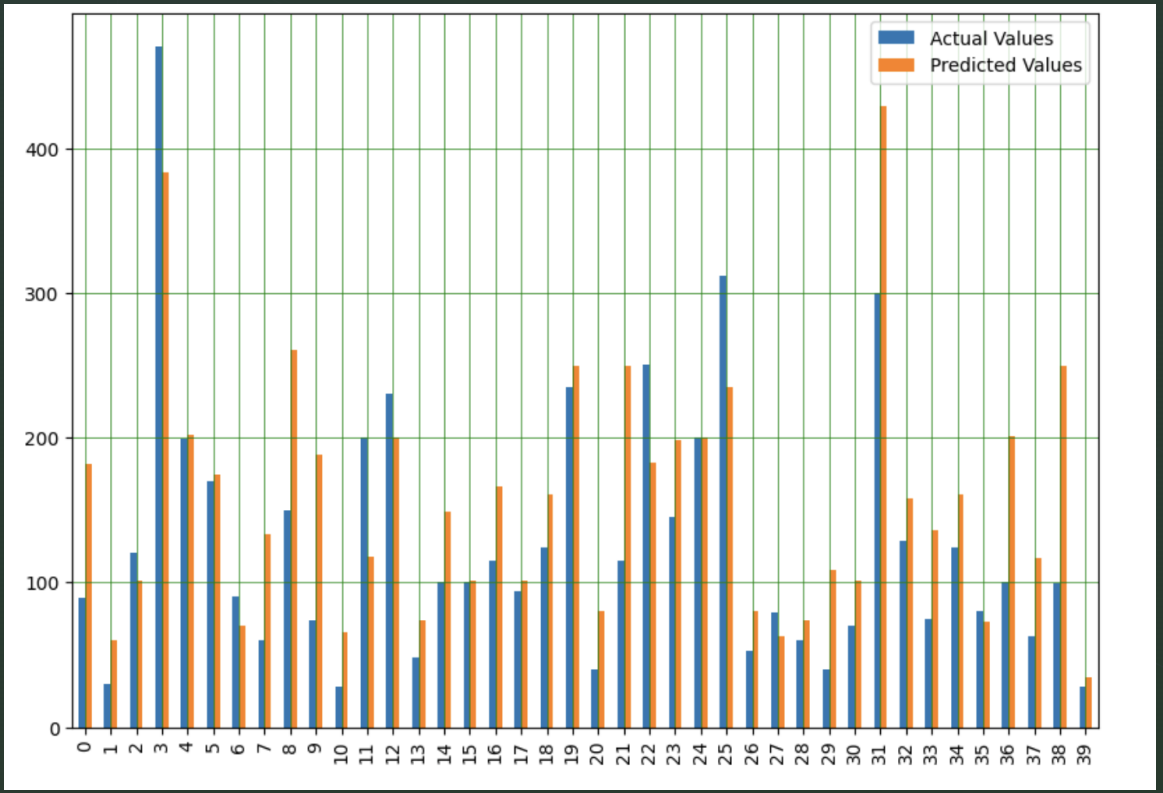
- **n_estimators – Количество деревьев (**сколько отдельных решающих деревьев будет построено в ансамбле**).** Начинаю с 100 и увеличиваю до тех пор, пока метрики (RMSE, R^2) не перестанут улучшаться.
- **max_depth – Глубина деревьев (максимальная глубина каждого дерева) то есть сколько «вопросов» задает каждое дерево**
- **min_samples_split – Минимальное число образцов для разделения узла (**сколько примеров должно быть в узле, чтобы его можно было разделить на поддеревья**)**
- **min_samples_leaf – Минимальное число образцов в листе (**Определяет минимальное количество примеров в конечном узле (листе).
- **max_features – Число признаков для разбиения узла (**сколько признаков рассматривается при каждом разбиении узла. Влияет на "разнообразие" деревьев**) будем брать треть признаков**
-

Ищем параметры случайным перебором

Лучшие параметры:
'max_depth’ = 7
'min_samples_split’ = 9
'n_estimators’ = 130
’min_sample_leaf’ = 4

Результаты применения случайного леса

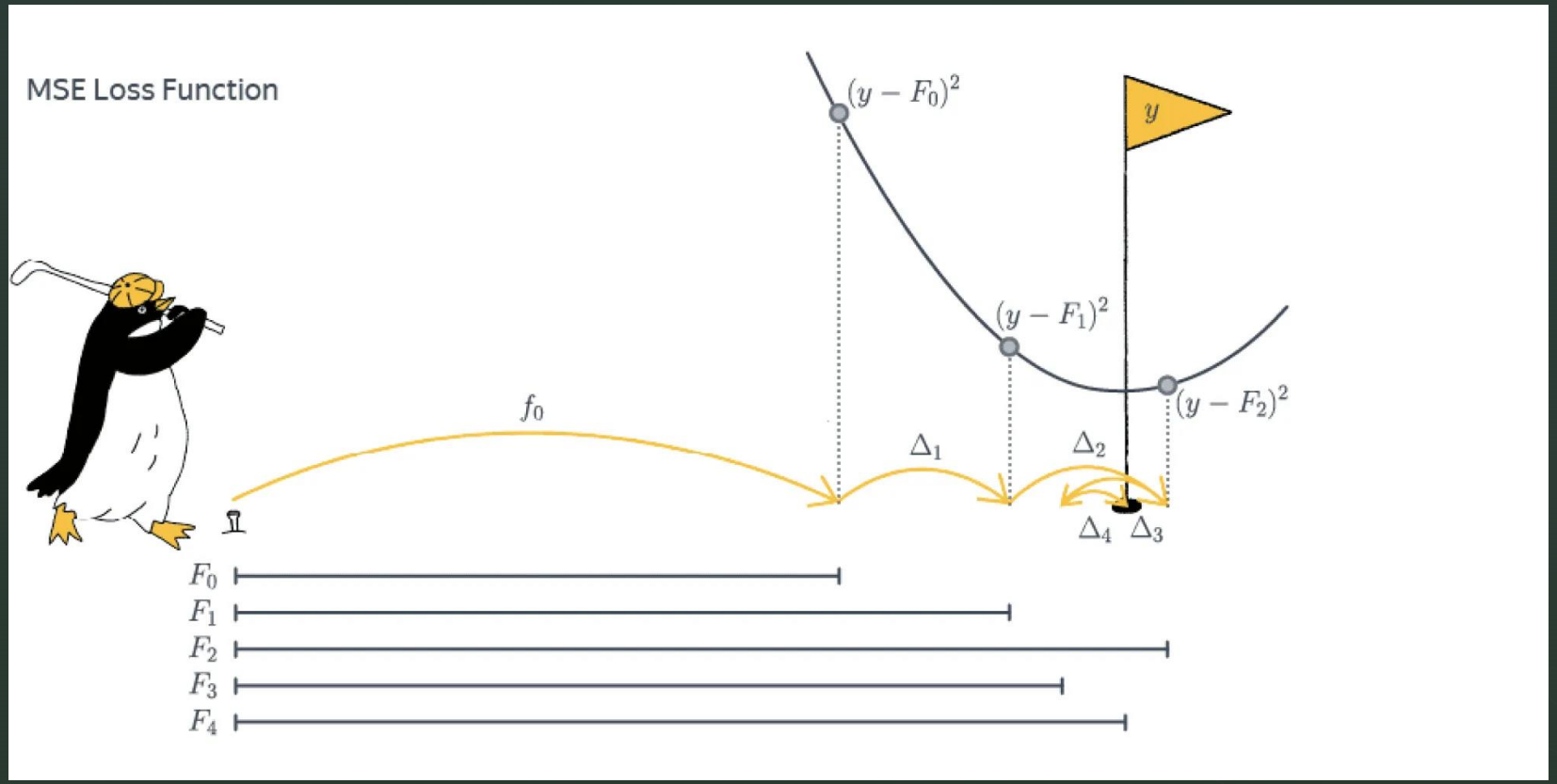
- Результат по метрикам: RMSE: 192.35 MAE: 66.45 R2 Score: 16.37%



3) градиентный бустинг

- Бустинг можно представить как гольфиста, цель которого — загнать мяч в лунку с координатой yballyball. Положение мяча здесь – ответ композиции a(xball)*a*(*x*ball). Гольфист мог бы один раз ударить по мячу, не попасть в лунку и пойти домой, но настырность заставляет его продолжить. Ему не нужно начинать каждый раз с начальной позиции. Следующий удар гольфиста переводит мяч из текущего положения ak(xball)*ak*(*x*ball) в положение ak+1(xball)*ak*+1(*x*ball). Каждый следующий удар — это та поправка, которую вносит очередной базовый алгоритм в композицию. Если гольфист все делает правильно, то функция потерь будет уменьшаться.

Глольфист

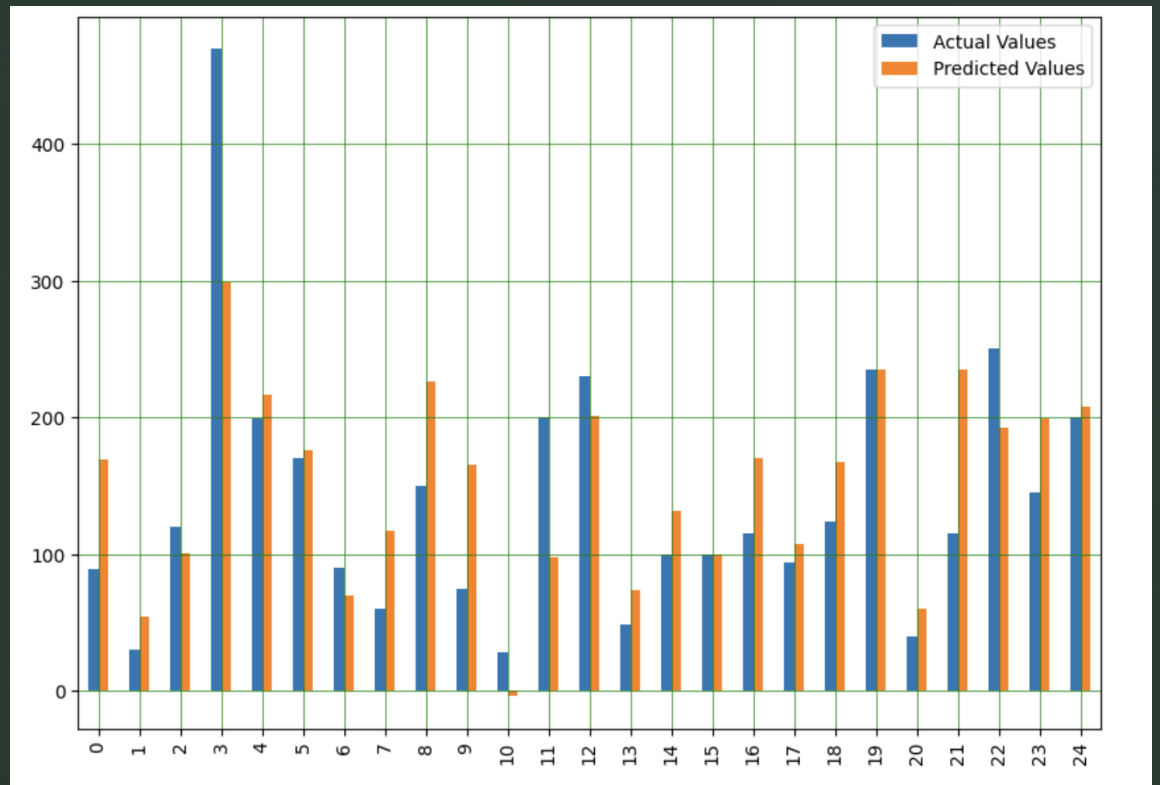

MSE Loss Function
$(y - F_0)^2$
y
$(y - F_1)^2$
$(y - F_2)^2$
f_0
Δ_1
Δ_2
Δ_4
Δ_3
F_0
F_1
F_2
F_3
F_4

Подбор параметров для градиентного бустинга

- 0) Параметры взятые наугад: n_estimators: 2000, learning_rate: 0.02
- 1)Лучшие параметры: {'learning_rate': 0.01687770422304368, 'n_estimators ': 2005}
- 2)Лучшие параметры: {'learning_rate': 0.084908023769472 5, 'n_estimators': 1230,
- 3)Лучшие параметры: {'learning_rate': 0.011104423424720 48, ' 'n_estimators ': 1622, '}

Результат применения градиентного бустинга

- Лучшими оказались параметры: `n_estimators: 2000, learning_rate: 0.02`
- `Метрики: RMSE: 191.95 MAE: 66.57 R2 Score: 16.71%`



Почему градиентный бустинг оказался лучше всех?

1) Потому что модель обучается на своих ошибках
2) Потому что модель использует градиентный спуск ,

что даёт возможность оптимизировать любую дифференцируемую функцию потерь