Лабораторная работа №3 **Подбор гиперпараметров модели**

Dataset: Электрички

Задание

0. Выбрать модель для обучения (Desicion tree, Random forest, SVM, KNN, Boosting)
1. Показать какие гиперпараметры есть у выбранной модели (В виде таблички).
2. Выбрать датасет для обучения и взависимости от модели подготовить данные
3. Подобрать гиперпараметры для модели и сравнить лучшие подборы, для (Grid Search, RandomSearch, Optuna)
4. На самом лучшем обучении (Grid Search, RandomSearch, Optuna) сделать калькулятор, который показывает локальную интерпретацию с помощью LIME и глобальную интерпретацию с помощью SHAP.

Исходные данные

Датасет: Записи о регистрации электромобилей на уровне VIN и атрибуты транспортных средств

Каждая строка описывает зарегистрированный электромобиль (или подключаемый гибрид) и включает частичный VIN (VIN (1-10)), год выпуска, марку и модель, тип электромобиля (BEV или PHEV), запас хода на электротяге (мили), базовую рекомендованную розничную цену (MSRP), примечания о соответствии требованиям для автомобилей на чистом альтернативном топливе (CAFV), геокодированное местоположение автомобиля (геометрия POINT), электроэнергетическую компанию и участок переписи 2020 года

Выбор модели

Для выполнения задания была выбрана модель XGBoost (Extreme Gradient Boosting) — продвинутая реализация градиентного бустинга,
где каждое следующее дерево учится на ошибках предыдущих.

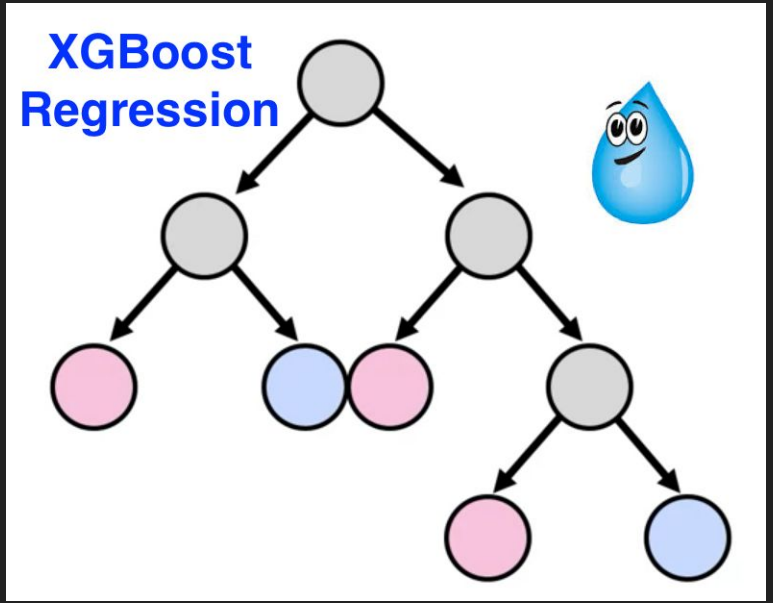
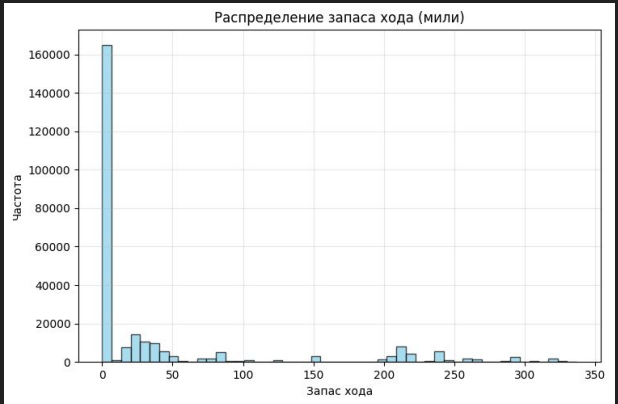
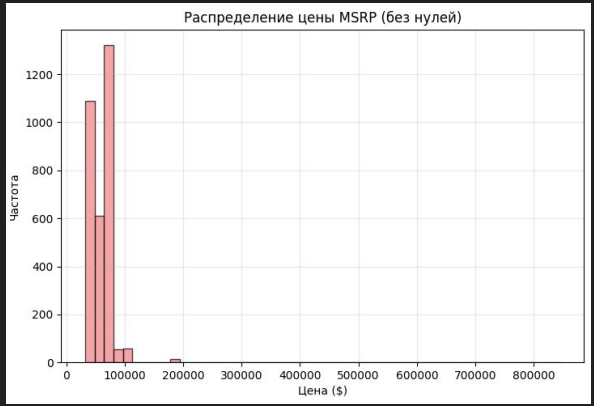


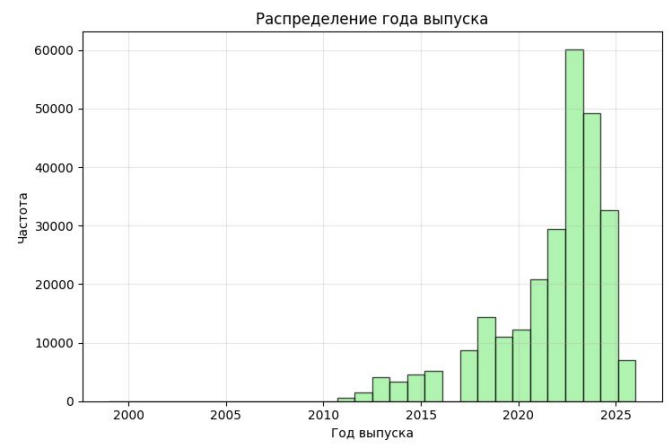
Таблица гиперпараметров

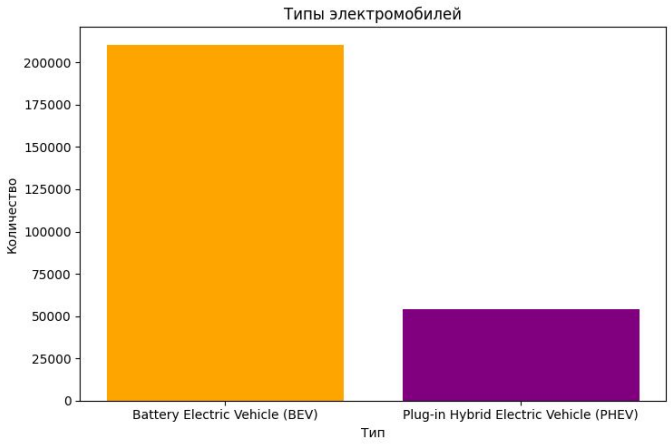
Гиперпараметр	Описание	Возможные значения / Диапазон
n_estimators	Количество деревьев в ансамбле	int: 50-1000
max_depth	Максимальная глубина дерева	int: 3-12 (None для без ограничений)
learning_rate	Скорость обучения (шаг градиентного спуска)	float: 0.01-0.3
subsample	Доля образцов для обучения каждого дерева	float: 0.5-1.0
colsample_bytree	Доля признаков для каждого дерева	float: 0.5-1.0
gamma	Минимальное уменьшение потерь для разделения узла	float: 0-10
reg_alpha	L1 регуляризация (Lasso)	float: 0-10
reg_lambda	L2 регуляризация (Ridge)	float: 0-10

Визуализация распределений

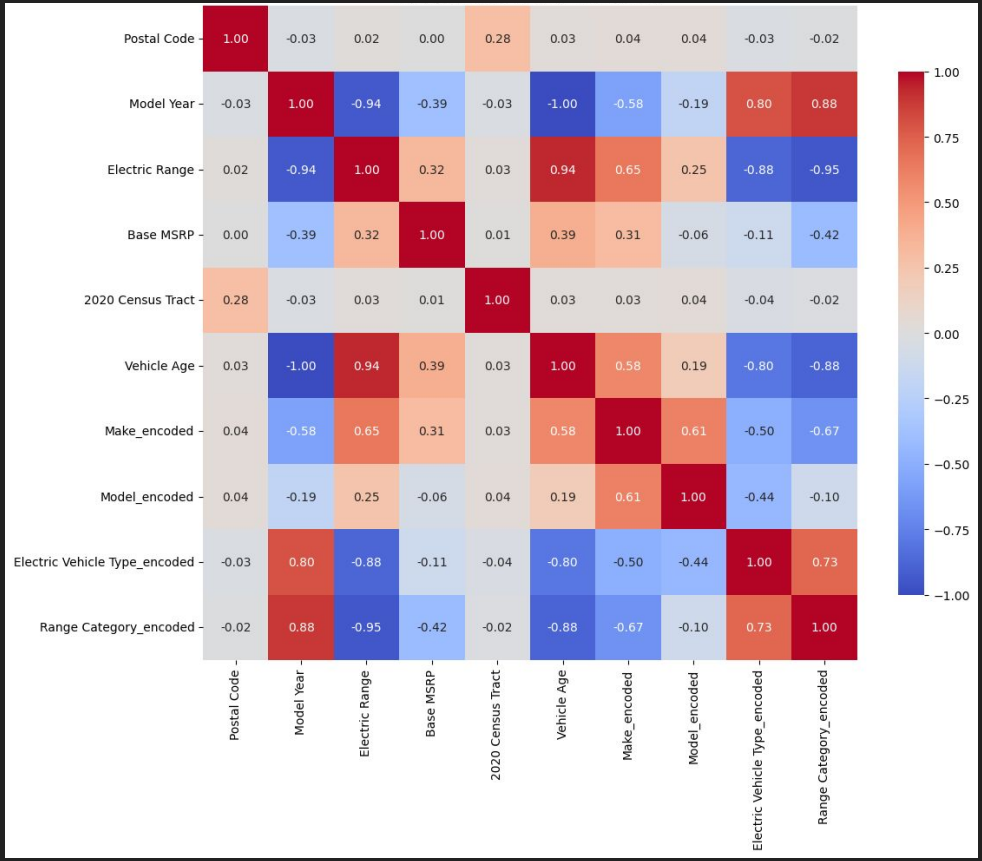








Корреляционная матрица



Grid Search

- Перебирает все возможные комбинации указанных гиперпараметров.
- Для каждой комбинации обучает модель и проверяет качество на кросс-валидации.
- Возвращает лучшую комбинацию по заданной метрике (например, accuracy).

```
Fitting 3 folds for each of 108 candidates, totalling 324 fits
Лучший R²: 0.9868
Лучшие параметры: {'colsample_bytree': 0.8, 'learning_rate': 0.05, 'max_depth': 6, 'n_estimators': 300, 'subsample': 0.8}
```

Random Search

Выбираем случайные комбинации гиперпараметров из заранее определённых диапазонов.

Плюсы:

* Прост в реализации.
* Эффективнее, чем Grid Search для моделей с большим числом гиперпараметров.
* Хорош для грубой оценки влияния гиперпараметров.

Минусы:

* Не гарантирует нахождение оптимального решения.
* Требует большого числа итераций при сложных пространствах гиперпараметров.

```
Fitting 3 folds for each of 50 candidates, totalling 150 fits
Лучший R²: 0.9920
Лучшие параметры:
colsample_bytree: 0.6390688456025535,
gamma: 6.842330265121569, learning_rate: 0.1420457481218804, max_depth: 9, n_estimators: 363,
reg_alpha: 0.34388521115218396, reg_lambda: 9.093204020787821, subsample: 0.7035119926400067,
```

Байесовская оптимизация с Optuna

строим вероятностную модель функции потерь и используем её, чтобы выбирать новые гиперпараметры «умнее», чем случайно.

Плюсы:

* Эффективнее Random Search при сложных пространствах гиперпараметров.
* Балансирует **exploration** (исследование) и **exploitation** (использование лучших областей).

Минусы:

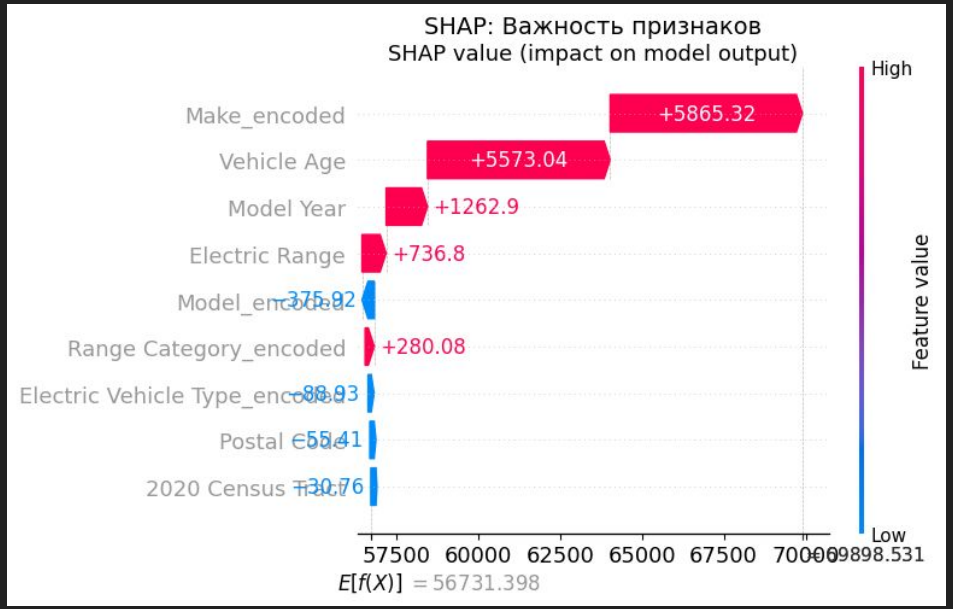
* Сложнее для понимания и настройки.
* Требует больше зависимостей (Optuna, иногда PyTorch/Scikit-learn).

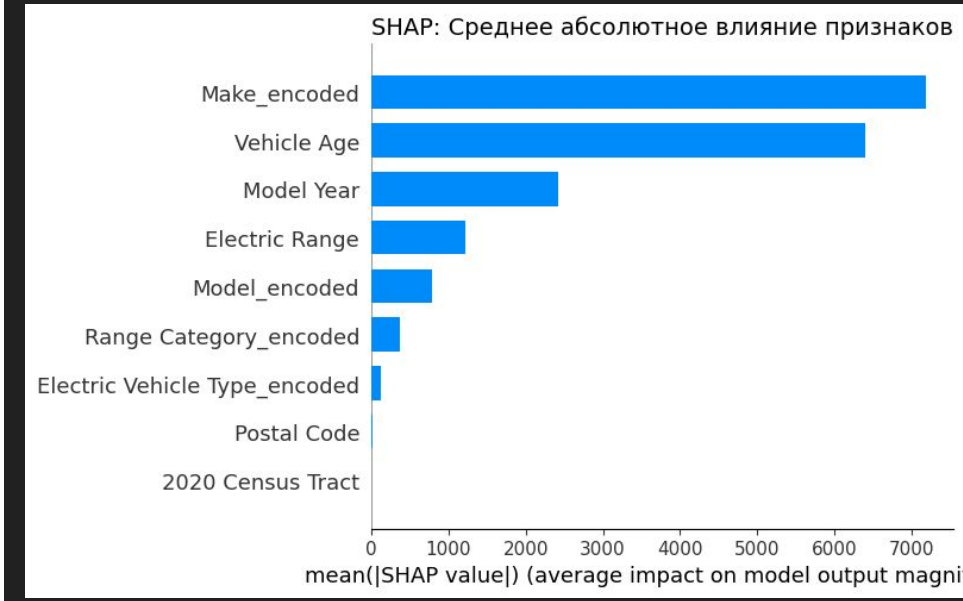
```
Лучший R²: 0.9923
Лучшие параметры: {'n_estimators': 192, 'max_depth': 4, 'learning_rate': 0.1153050640337842 8, 'subsample': 0.8609522266203155, 'colsample_bytree': 0.640745090259403,
                   'gamma': 2.82391605902394, 'reg_alpha': 0.09974351114088487, 'reg_lambda': 5.886835150474556
```

Интерпретация с LIME

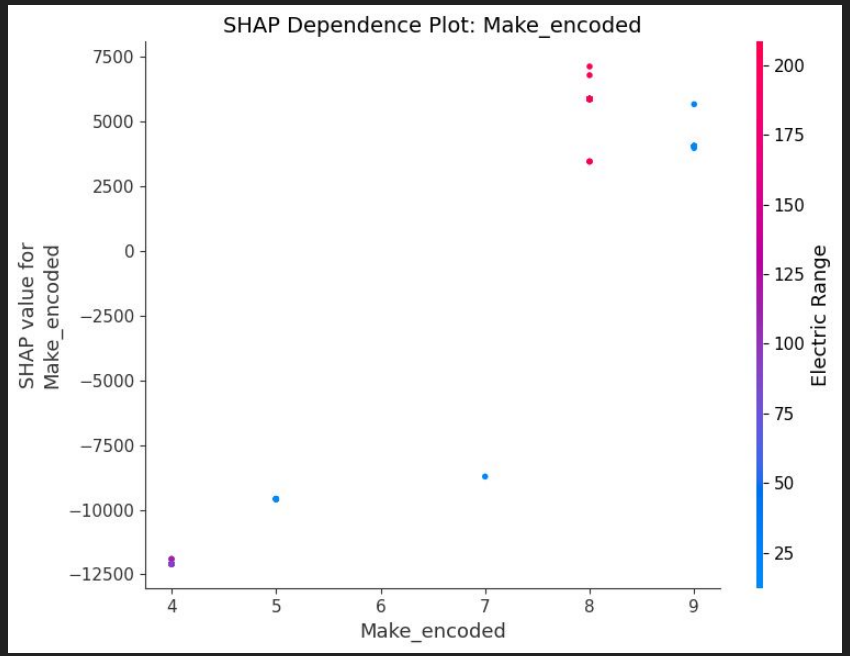
```
Пример #0:
Реальная цена: $69,900
Предсказанная цена: $69,899
Ошибка: $1

Объяснение предсказания (ТОП-10 признаков):
============================================================
7.00 < Vehicle Age <= 11.00              → -10214.6724
4.00 < Make_encoded <= 8.00              → -9567.4769
32.00 < Electric Range <= 208.00         → -5984.0065
Model_encoded <= 9.00                    → +2701.4954
Range Category_encoded <= 0.00           → +888.7170
2013.00 < Model Year <= 2014.00          → +778.1558
Electric Vehicle Type_encoded <= 0.00    → -443.2241
2020 Census Tract > 53053072408.00       → -18.8624
Postal Code > 98334.00                   → -7.4239

============================================================
Интерпретация:

Признаки, УВЕЛИЧИВАЮЩИЕ предсказание:
  • Model_encoded <= 9.00: +2701.495
  • Range Category_encoded <= 0.00: +888.717
  • 2013.00 < Model Year <= 2014.00: +778.156
...
  • 4.00 < Make_encoded <= 8.00: -9567.477
  • 32.00 < Electric Range <= 208.00: -5984.007
  • Electric Vehicle Type_encoded <= 0.00: -443.224
  • 2020 Census Tract > 53053072408.00: -18.862
```

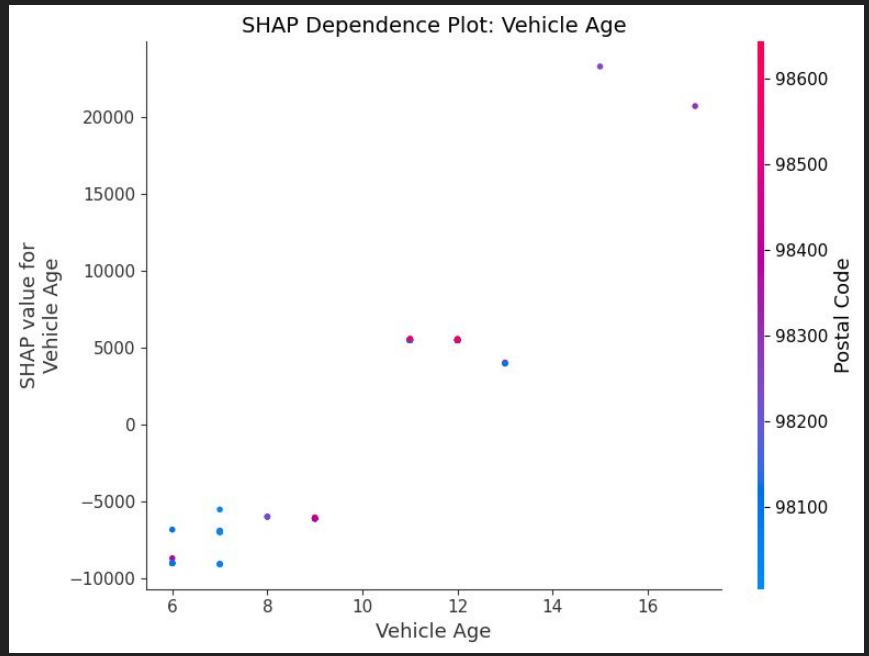
SHAP интерпретация



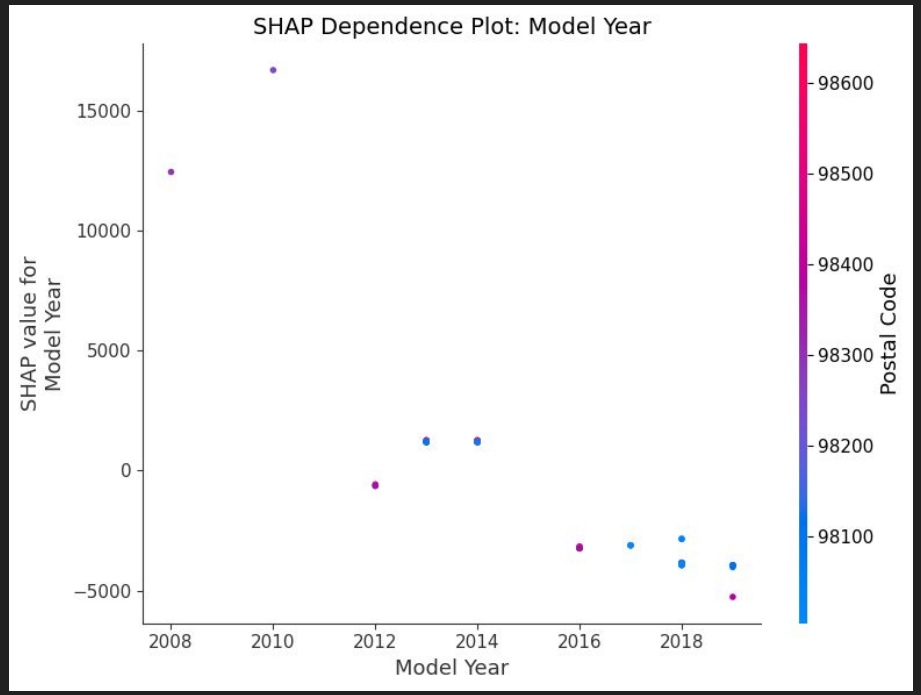


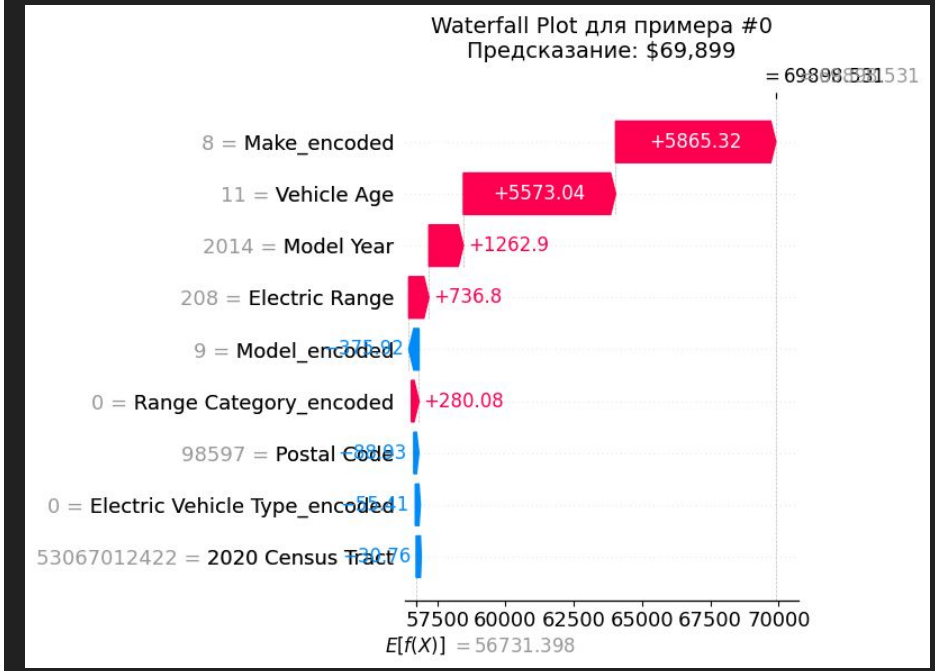
SHAP интерпретация



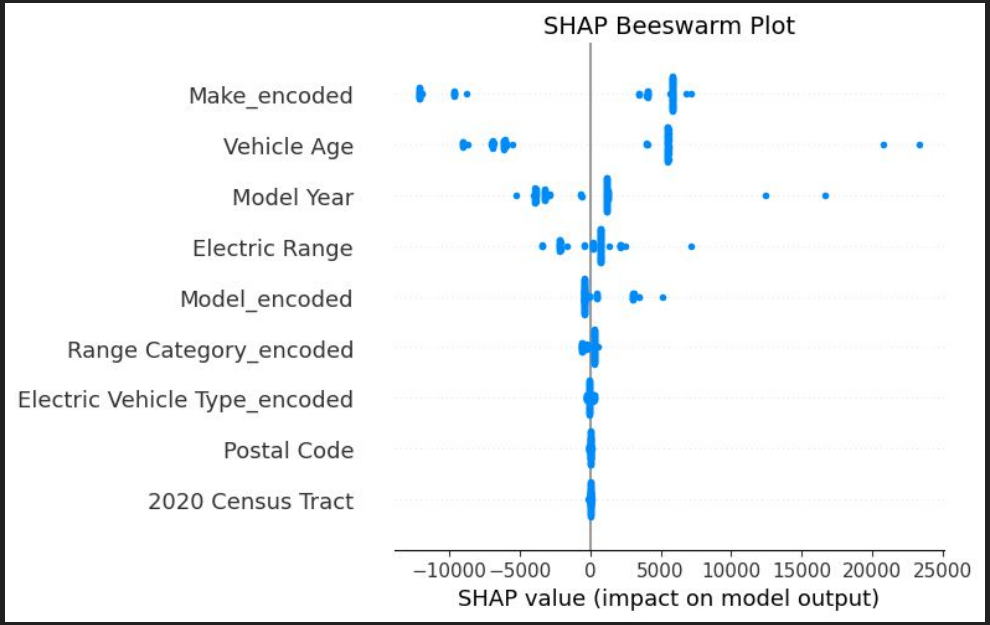


SHAP интерпретация





SHAP интерпретация



Базовое значение (expected value): $56,731
Среднее абсолютное влияние по признакам:

Признак	SHAP_важность
Make_encoded	7178.656738
Vehicle Age	6395.799805
Model Year	2426.657715
Electric Range	1212.933960
Model_encoded	795.915466
Range Category_encoded	372.194458
Electric Vehicle Type_encoded	121.294571
Postal Code	18.901405
2020 Census Tract	6.782192