УДК 004

Разработка системы предсказания цены телевизоров на основе технологий машинного обучения

Быстрова Ольга Анатольевна
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
Целью исследование является применение различных методов машинного обучения для построения регрессионной модели цены на телевизоры. В результате исследования выбрана модель и переменные, обеспечивающие лучшие результаты для текущей задачи. Сравнение моделей произведено на основании основных метрик: R2, RMSE, MAE, MSE.
Ключевые слова: модель обучения, датасет, авито.

Development of a monitor price prediction system based on machine learning technologies

Bystrova Olga Anatolyevna
Sholom-Aleichem Priamursky State University
Student

Abstract
This article describes the learning process of a system that predicts the price of a TV based on characteristics using its own dataset. As a result of the research, the model and variables that provide the best results for the current task were selected. The models were compared based on the main indicators: R2, RMSE, MAE, MSE.
Keywords: learning model, dataset, avito.

1 Введение

1.1 Актуальность

В современном мире объемы данных растут в геометрической прогрессии, и умение извлекать ценную информацию становится все более критическим для успешного функционирования бизнеса. Одной из важнейших задач в этом контексте является анализ и понимание факторов, влияющих на ценообразование товаров и услуг.

1.2Обзор исследований

Е.М. Аксютина и Ю.С. Белов в своей статье рассматривают основные архитектуры, применяемые для анализа больших данных, их особенности и ограничения, а так же выявляются требования к методам машинного обучения, выполнение которых позволит применять их при анализе больших данных [1].

К.А. Найденова и О.А. Невзорова рассматривают современные методы машинного обучения, применяемые в задачах обработки естественного языка [2].

Б.И. Гельцер, М.М. Циванюк представили анализ научной литературы по результатам использования методов машинного обучения [3].

В.С. Старостин Рассмотрел цифровую трансформацию маркетинга [4].

В.И. Донской провел сравнительный анализ различных определений обучаемости, рассмотрены необходимые и достаточные условия обучаемости, указаны границы применимости VC теории [5].

1.3Цель исследования

Цель исследования является проведение исследования и анализа данных о телевизорах, а также разработка и оценка эффективности моделей машинного обучения для данного датасета.

2 Материалы и методы

Для реализации использовались среды разработки GoogleColabи Pycharm, а также библиотек и регрессионных моделей: LinearRegression, PolynomialFeatures, DecisionTree, RandomForest, KNeighbors, иXGBOOST.

Датасет был создан при помощи парсинга. Ссылка на датасет: https://colab.research.google.com/drive/1Vv_C5lznj91DoNphclqnTB6UZwoQTxRp.

3 Результаты и обсуждения

Приведем этапы исследования.

Загрузка данных (Рис. 1).

```
# Загрузка данных
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Загрузка данных из файла televisions.xlsx
df = pd.read_excel("televisions.xlsx")
```

```
# Просмотр первых нескольких строк данных
print(df.head())
```

```
       Name         model  price     log10  diagonal display technology  \
0      Sony  KDL-40V4000   6000  3.778151        32                LED
1     Океан   LC-42W2101   8499  3.929368        19                LCD
2  Samsung     T27D390EX   6000  3.778151        26                LED
3        LG     42LN540V   8000  3.903090        22                LCD
4    Xiaomi   MI TVA2 43  21500  4.332438        32               OLED
```

Рисунок 1 –Загрузка данных

```
print(df.describe())
```

	price	log10	diagonal	number of ports	year
count	199.000000	199.000000	199.000000	199.000000	199.000000
mean	11975.040201	3.824460	35.216080	3.603015	2014.919598
std	18349.391082	0.464360	13.696882	1.072235	4.685413
min	500.000000	2.698970	19.000000	2.000000	2006.000000
25%	3000.000000	3.477121	26.000000	3.000000	2010.000000
50%	7000.000000	3.845098	32.000000	3.000000	2016.000000
75%	12000.000000	4.079181	42.000000	4.000000	2019.000000
max	185000.000000	5.267172	105.000000	6.000000	2027.000000

Рисунок 2 – Анализ данных

Затем выполняем предобработку данных для моделирования.

```
# Кодирование категорий
df = pd.get_dummies(df, columns=['display technology'])

# Обновим X и y для эксперимента с использованием цены (price)
X_price = df[['diagonal', 'year']]
y_price = df['price']
```

Рисунок 3 – Предобработка данных

Далее выполняем разделение данных на обучающий и тестовый набор. Создаем модель линейной регрессии и обучаем ее на обучающих данных.

```
X_train_single, X_test_single, y_train, y_test = train_test_split(X_single, y, test_size=0.2, random_state=42)

# Обучение модели (одна фича)
model_single = LinearRegression()
model_single.fit(X_train_single, y_train)
```

▾ LinearRegression
LinearRegression()

Рисунок 4 – Обучение модели

Выполняем предсказание целевой переменной с использованием обученной модели на тестовом наборе данных.

```
y_pred_single = model_single.predict(X_test_single)
mse_single = mean_squared_error(y_test, y_pred_single)
mae_single = mean_absolute_error(y_test, y_pred_single)
rmse_single = np.sqrt(mse_single)
r2_single = r2_score(y_test, y_pred_single)
```

```
print("MSE:", mse_single)
print("MAE:", mae_single)
print("RMSE:", rmse_single)
print("R2 Score:", r2_single)
```

```
MSE: 81739158.97645012
MAE: 7090.435083075189
RMSE: 9040.971130163512
R2 Score: 0.5965650618545619
```

Рисунок 5 – Вывод метрик

Создадим модель случайного леса для решения задачи. Импортируем соответствующий класс. Затем создаем экземпляр модели с заданным параметром, который обеспечивает воспроизводимость результатов.

```
model_rf_double = RandomForestRegressor(random_state=42)
model_rf_double.fit(X_train_double, y_train)
```

```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

```
# Вывод метрик для модели Random Forest (две фичи)
y_pred_rf_double = model_rf_double.predict(X_test_double)
mse_rf_double = mean_squared_error(y_test, y_pred_rf_double)
mae_rf_double = mean_absolute_error(y_test, y_pred_rf_double)
rmse_rf_double = np.sqrt(mse_rf_double)
r2_rf_double = r2_score(y_test, y_pred_rf_double)
```

```
print("MSE:", mse_rf_double)
print("MAE:", mae_rf_double)
print("RMSE:", rmse_rf_double)
print("R2 Score:", r2_rf_double)
```

```
MSE: 2038793.1623775016
MAE: 301.4702500000002
RMSE: 1427.863145535139
R2 Score: 0.9899372540205351
```

Рисунок 6 – Модель дерева решений

Модель k-ближайших соседей (KNN) для задачи регрессии. Код выполняет предсказание целевой переменной на тестовом наборе данных, и вычисляются метрики оценки качества модели.

```
model_knn_single = KNeighborsRegressor()
model_knn_single.fit(X_train_single, y_train)
```

```
▾ KNeighborsRegressor
KNeighborsRegressor()
```

```
y_pred_knn_single = model_knn_single.predict(X_test_single)
mse_knn_single = mean_squared_error(y_test, y_pred_knn_single)
mae_knn_single = mean_absolute_error(y_test, y_pred_knn_single)
rmse_knn_single = np.sqrt(mse_knn_single)
r2_knn_single = r2_score(y_test, y_pred_knn_single)
```

```
print("MSE:", mse_knn_single)
print("MAE:", mae_knn_single)
print("RMSE:", rmse_knn_single)
print("R2 Score:", r2_knn_single)
```

```
MSE: 4042338.416999998
MAE: 438.6949999999999
RMSE: 2010.556743044075
R2 Score: 0.9800484789708299
```

Рисунок 7 –Модели k-ближайших соседей

Обучим последнюю модель XGBoost. Затем выполняется предсказание целевой переменной на тестовом наборе данных, и вычисляются метрики оценки качества модели.

```
model_xgb_single = XGBRegressor(random_state=42)
model_xgb_single.fit(X_train_single, y_train)
```

```
▸ XGBRegressor
```

```
y_pred_xgb_single = model_xgb_single.predict(X_test_single)
mse_xgb_single = mean_squared_error(y_test, y_pred_xgb_single)
mae_xgb_single = mean_absolute_error(y_test, y_pred_xgb_single)
rmse_xgb_single = np.sqrt(mse_xgb_single)
r2_xgb_single = r2_score(y_test, y_pred_xgb_single)
```

```
print("MSE:", mse_xgb_single)
print("MAE:", mae_xgb_single)
print("RMSE:", rmse_xgb_single)
print("R2 Score:", r2_xgb_single)
```

```
MSE: 11418234.330847489
MAE: 748.3252990722656
RMSE: 3379.0877956702293
R2 Score: 0.943643723293963
```

Рисунок 8 –Модель XGBoost

Логарифмирование цены

```
y = np.log2(df['Cena'])
```

Рисунок 9 – Логарифмирование цены

Построение моделей с прологарифмированной ценой (дерево)

```
[ ] tree = DecisionTreeRegressor(max_depth=5)
    tree.fit(X_train, y_train)
    training_metrics(tree, X_train, y_train)
```

```
train metrics
----------
Score: 0.9214994599349471
RMSE: 0.23711182880349096
MAE: 0.16503386599708236

Cross Validation
----------

Mean CV score: 0.26758258067323804
Standart deviation: 0.03902366465890646
```

Рисунок 10 – дерево, прологарифмированная цена

```
[ ] forest = RandomForestRegressor(max_depth=5, n_jobs=-1)
    forest.fit(X_train, y_train.ravel())
    training_metrics(forest, X_train, y_train.ravel())
```

```
train metrics
----------
Score: 0.9197207617460179
RMSE: 0.2397830716990936
MAE: 0.1679248300851951

Cross Validation
----------

Mean CV score: 0.2701964244115702
Standart deviation: 0.04377097211463712
```

Рисунок 11 –лес, прологарифмированная цена

```
[ ] knn_regressor = KNeighborsRegressor(n_neighbors=5)
    knn_regressor.fit(X_train, y_train)
    training_metrics(knn_regressor, X_train, y_train)
```

```
train metrics
----------
Score: 0.8638570714035358
RMSE: 0.31225862439157226
MAE: 0.24212066922730788

Cross Validation
----------

Mean CV score: 0.31036350162910364
Standart deviation: 0.03993358174713914
```

Рисунок 12 – KNN, прологарифмированная цена

```
[ ]  lin_reg = LinearRegression()
     lin_reg.fit(X_train, y_train)
     training_metrics(lin_reg, X_train, y_train)

     train metrics
     ----------
     Score: 0.8754615594795357
     RMSE: 0.29865418817713896
     MAE: 0.23773549669876112

     Cross Validation
     ----------

     Mean CV score: 0.2982798269203829
     Standart deviation: 0.039178618632369044
```

Рисунок 13 – Линейная регрессия, прологарифмированная цена

Выводим итоговую таблицу 1.

Таблица 1. Сравнение точности предугадывания результата анализа при различных методах машинного обучения

Метод	Точность получившихся моделей (%) Обычная цена	Точность получившихся моделей (%) Цена лог10
RandomForest	88%	93%
DecisionTree	84%	88%
KNN	79%	92%
XGBoost	87%	89%
CatBoostRegressor	87%	89%

Из таблицы видно, что RandomForest показал наилучшую производительность с наименьшей ошибкой прогноза. Он эффективно обрабатывает категориальные переменные и может автоматически обрабатывать пропущенные значения. Кроме того, RandomForest обладает высокой производительностью и способностью обобщать данные.

Выводы

Таким образом, на основе проведенного анализа и экспериментов, можно рекомендовать использовать RandomForest для прогнозирования цены на телевизоры. Однако, для дальнейшего улучшения модели возможно потребуется дополнительное исследование и настройка параметров.

Библиографический список

1. Аксютина Е. М., Белов Ю. С. Обзор архитектур и методов машинного обучения для анализа больших данных //Электронный журнал: наука,

техника и образование. 2016. №. 1. С. 134-141.
2. Донской В. И. Машинное обучение и обучаемость: сравнительный обзор //IntellectualArchive. 2012. №. 933-19.
3. Найденова К. А., Невзорова О. А. Машинное обучение в задачах обработки естественного языка: обзор современного состояния исследований //Ученые записки Казанского университета. Серия Физико-математические науки. 2008. Т. 150. №. 4. С. 5-24.
4. Гельцер Б. И. и др. Методы машинного обучения в оценке предтестовой вероятности обструктивных и необструктивных поражений коронарного русла //Российский кардиологический журнал. 2020. №. 5. С. 99-105.
5. Старостин В. С. Трансформация маркетинговых технологий в эпоху машинного интеллекта //Вестник университета. 2018. №. 1. С. 28-34.