



Москва

23 августа

2025

Прогнозирование продаж товаров категории FMCG на основе ежедневных данных с помощью методов машинного обучения

Выполнила
DS-16 В.И. Вербицкая
Научный руководитель
Е.А. Паточенко



Основные положения

Цель исследования: Разработать надежную модель, предсказывающую продажи товаров, учитывающую сезонные колебания, наличие товаров на складе, цены и промоакции. Модель должна демонстрировать хорошую обобщающую способность как во времени, так и по категориям.

Предмет исследования: Продажи товаров категории FMCG в регионах Польши

Подход:

- Разведывательный анализ (EDA)
- Предиктивный анализ
- Предложения по дальнейшему развитию исследования

Результаты исследования могут быть полезны при планировании продаж, управлении запасами и организации маркетинговых акций



Данные

Источник данных: [Kaggle - FMCG Daily Sales Data 2022-2024](#)

Набор синтетических данных о ежедневных продажах товаров повседневного спроса (FMCG)

Размер датасета: **190 757** строк x **14** колонок

Временной интервал: с **21.01.2022** по **31.12.2024**

Целевой показатель: **units_sold (int)**

Нет пропущенных значений

Нет повторяющихся строк

Есть отрицательные значения: **stock_available, delivered_qty, units_sold**



Переменная

Описание

date	дата продажи
sku	идентификатор товара
brand	бренд товара
segment	сегмент товара
category	категория товара
channel	канал сбыта
region	регион продажи
pack_type	тип упаковки
price_unit	цена ед. товара
promotion_flag	флаг промоакции
delivery_days	доставка, дн.
stock_available	доступный запас, ед.
delivered_qty	доставлено, ед.
units_sold	продано, ед

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 190757 entries, 0 to 190756
```

```
Data columns (total 14 columns):
```

#	Column	Non-Null Count	Dtype
0	date	190757 non-null	object
1	sku	190757 non-null	object
2	brand	190757 non-null	object
3	segment	190757 non-null	object
4	category	190757 non-null	object
5	channel	190757 non-null	object
6	region	190757 non-null	object
7	pack_type	190757 non-null	object
8	price_unit	190757 non-null	float64
9	promotion_flag	190757 non-null	int64
10	delivery_days	190757 non-null	int64
11	stock_available	190757 non-null	int64
12	delivered_qty	190757 non-null	int64
13	units_sold	190757 non-null	int64

```
dtypes: float64(1), int64(5), object(8)
```

```
memory usage: 20.4+ MB
```

```
(190757, 14)
```

Необходимо перевести data из строкового типа в datetime

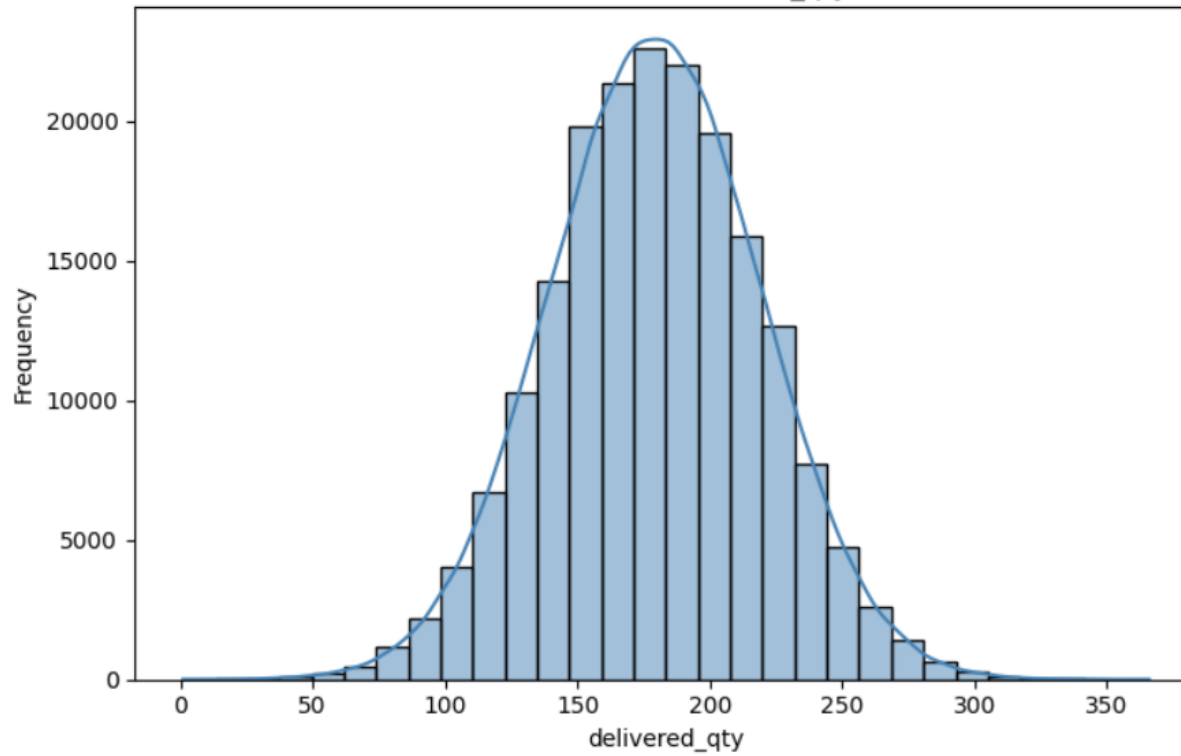


	date	price_unit	promotion_flag	delivery_days	stock_available	delivered_qty	units_sold
count	190757	190757.000000	190757.000000	190757.000000	190757.000000	190757.000000	190757.000000
mean	2023-10-28 04:11:33.514785536	5.251979	0.149200	3.004860	157.697652	179.333655	19.919709
min	2022-01-21 00:00:00	1.500000	0.000000	1.000000	-12.000000	-11.000000	-25.000000
25%	2023-04-16 00:00:00	3.380000	0.000000	2.000000	124.000000	152.000000	12.000000
50%	2023-11-12 00:00:00	5.250000	0.000000	3.000000	155.000000	179.000000	18.000000
75%	2024-06-07 00:00:00	7.130000	0.000000	4.000000	192.000000	206.000000	25.000000
max	2024-12-31 00:00:00	9.000000	1.000000	5.000000	405.000000	366.000000	139.000000
std	NaN	2.166705	0.356287	1.414626	52.736104	40.037475	11.770077

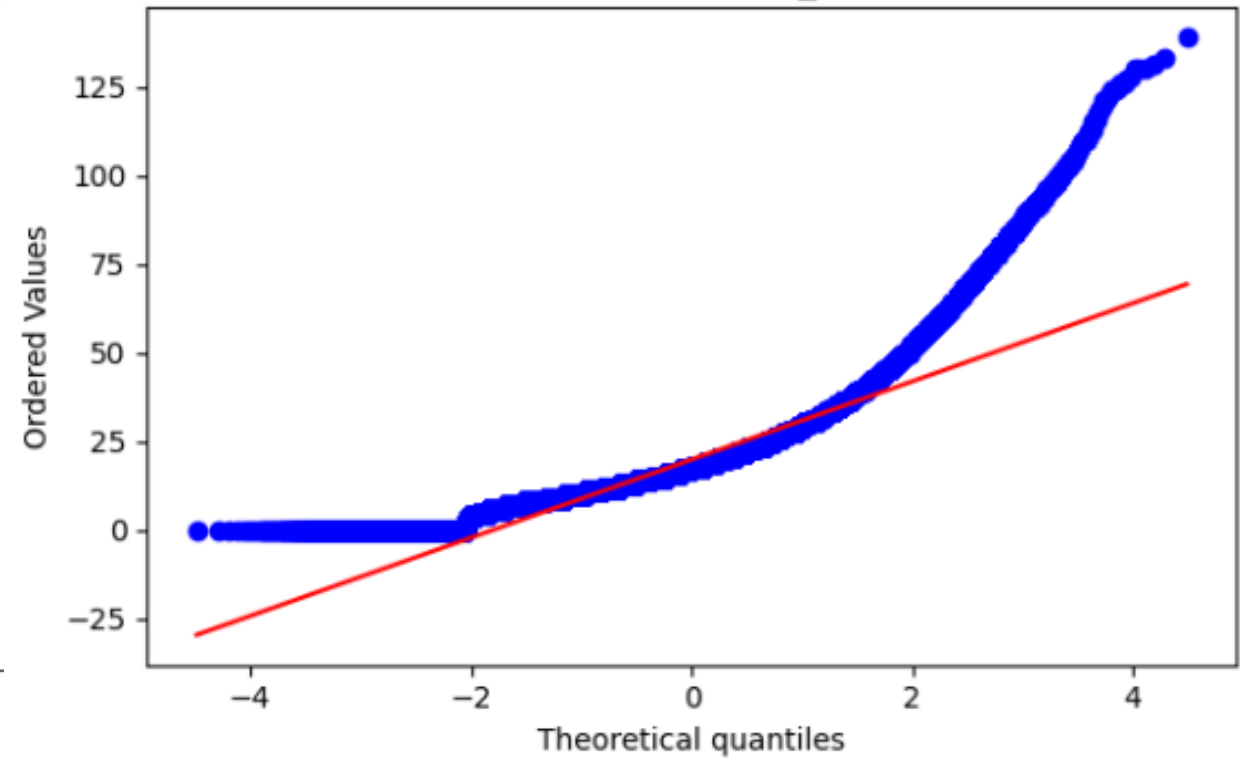
	sku	brand	segment	category	channel	region	pack_type
count	190757	190757	190757	190757	190757	190757	190757
unique	30	14	13	5	3	3	3
top	MI-006	SnBrand2	Yogurt-Seg1	Yogurt	Retail	PL-North	Carton
freq	8221	26775	26851	72707	63688	63645	63671



Distribution: delivered_qty



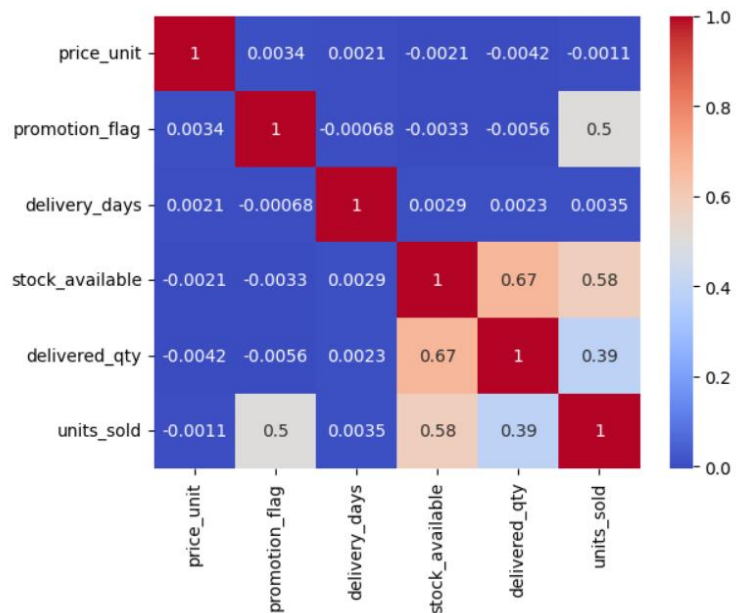
Q-Q Plot for units_sold





Москва

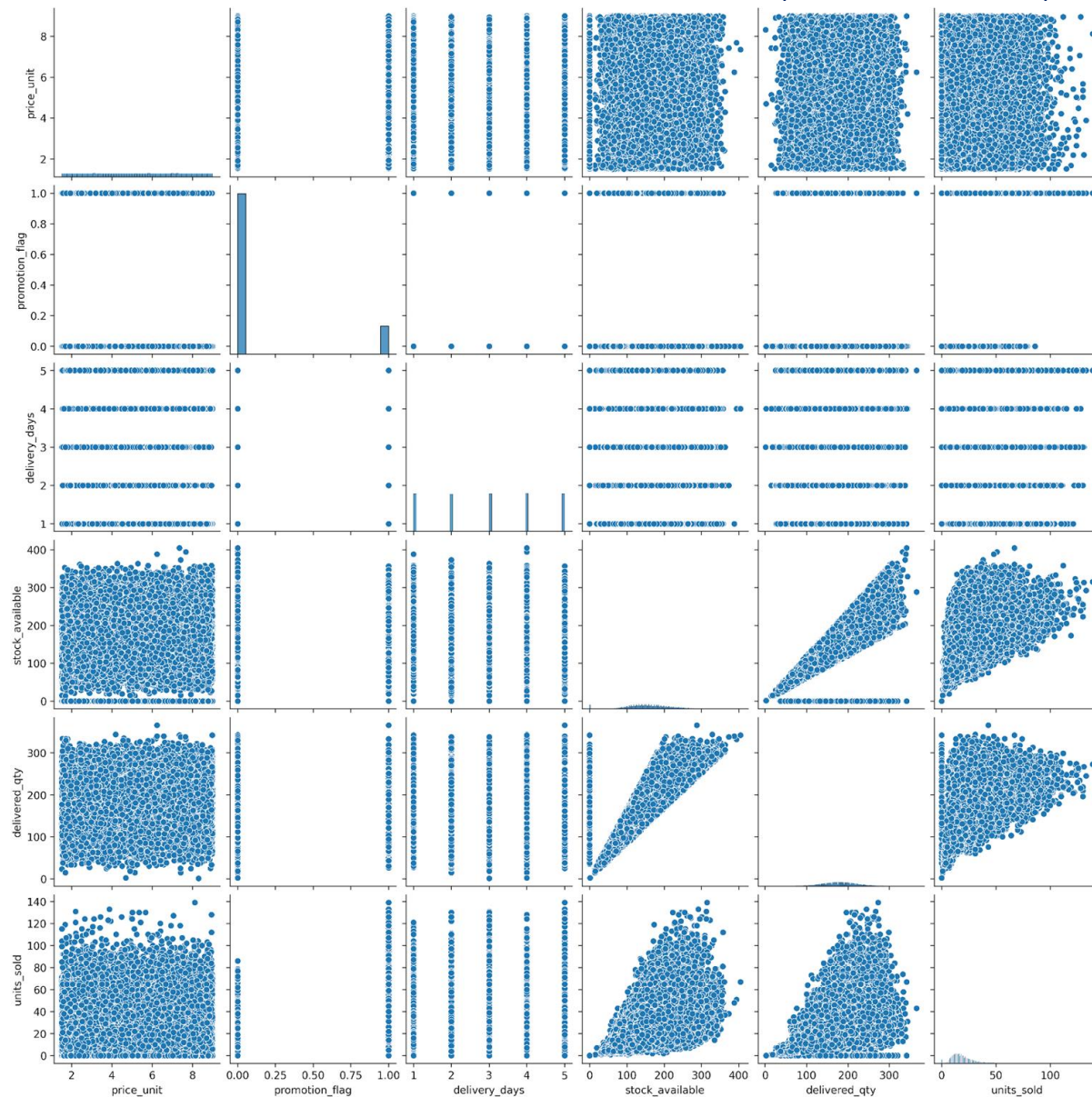
23 августа



Признак	Корреляция	Доменная значимость (+/-)	Визуальный тренд	Оставить?
price_unit	~ 0.0	+	?	❖ ?
promotion_flag	0.50	+	нет	❖ ?
delivery_days	0.01	+	нет	❖ ?
stock_available	0.58	+	положит.	✅ Да
delivered_qty	0.39	+	положит.	✅ Да

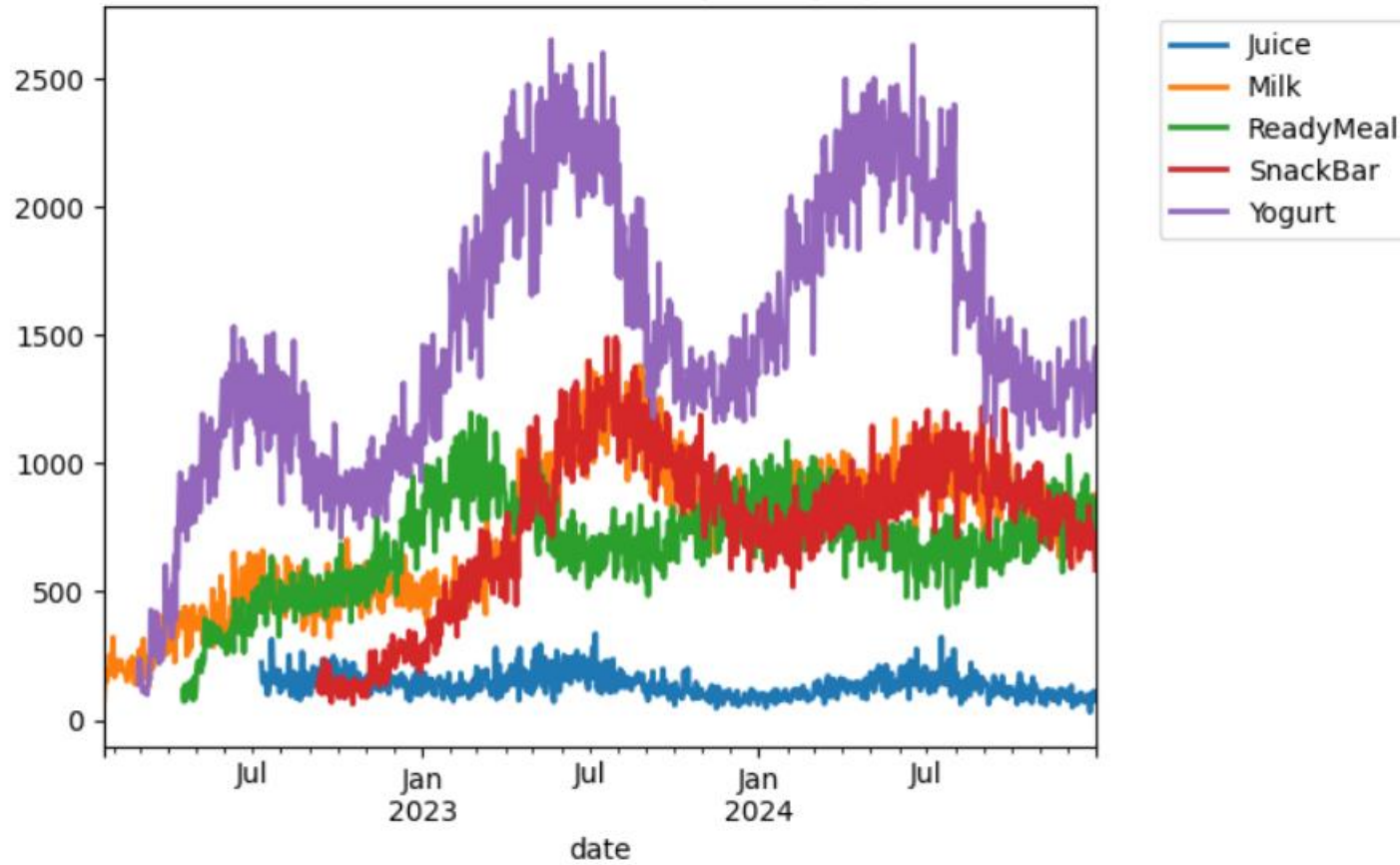
2025

7

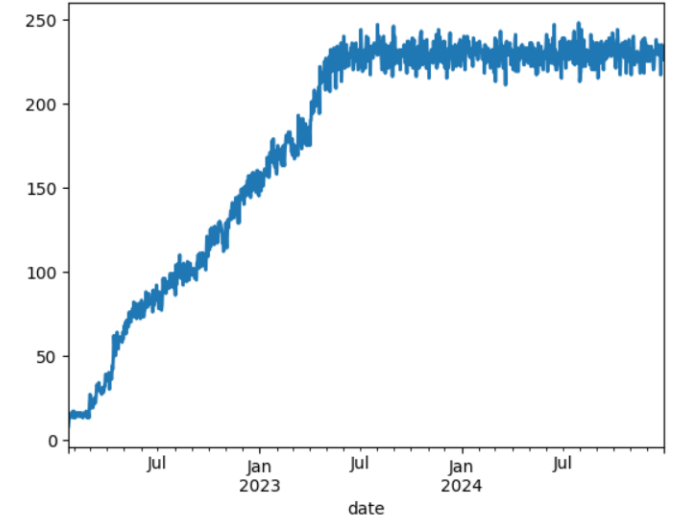




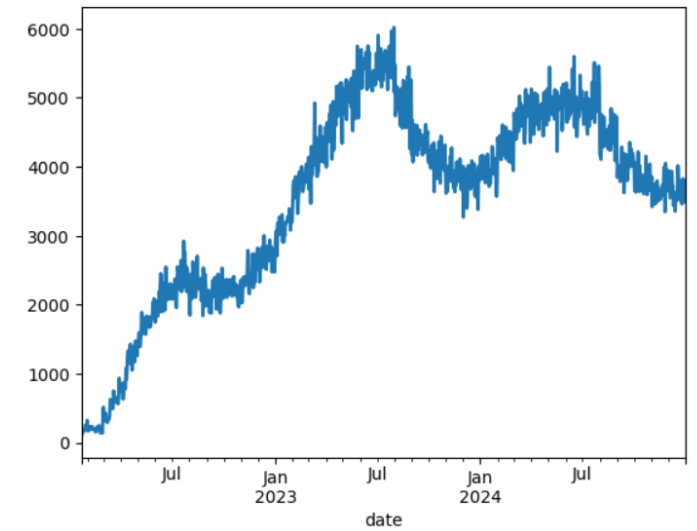
Units Sold Over Time by Category



Entries per Day

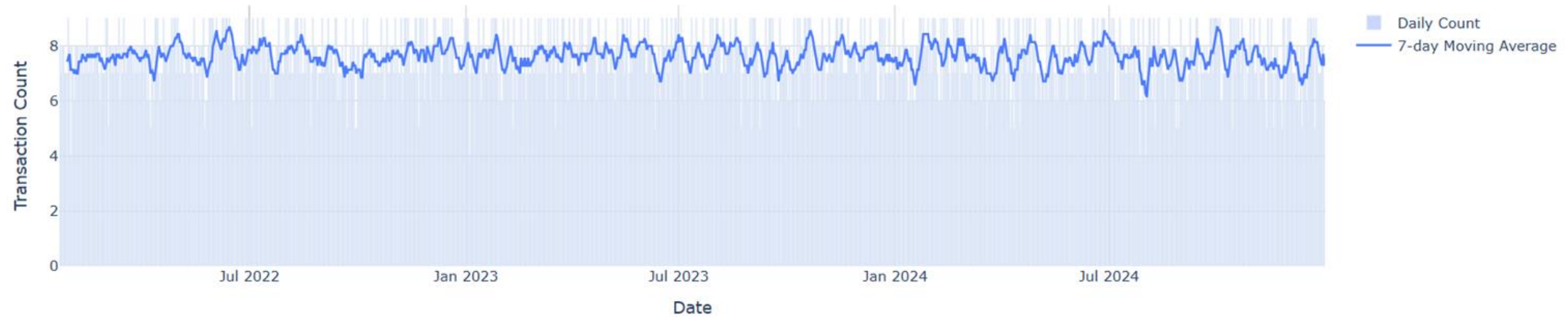


Total Units Sold over Time

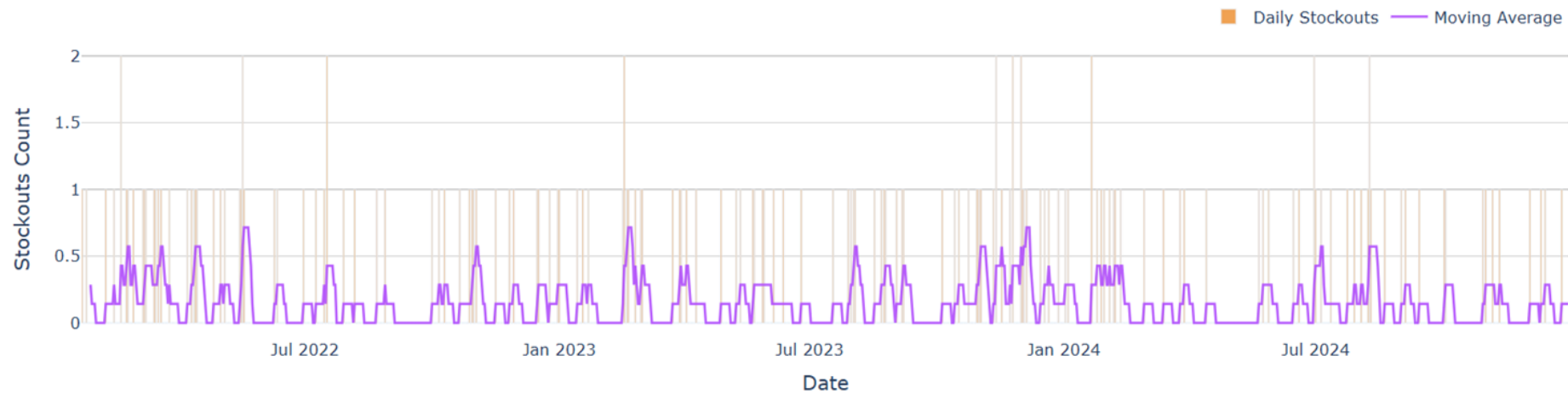




Daily Transaction Volume (Smoothed) — SKU: MI-006



Daily Events – SKU: MI-006





Новые переменные

1. Преобразование формата data из строкового типа в datetime
2. Преобразование категориальных переменных **'sku', 'segment', 'category'**
3. Создание новых переменных «календарь»: **'day', 'month', 'year', 'is_start_of_month', 'is_end_of_month', 'day_of_week', 'is_weekend', 'week_of_month'**
4. Кодирование запаздываний: лаг целевой переменной - **lag_1, lag_2, lag_3, lag_7, lag_14, lag_28**, **lag_1, lag_2, lag_3, lag_7, lag_14, lag_28** – лаг целевой переменной, **'momentum_7_1', 'momentum_14_7'** – коэффициент запаздывания, **'price_lag_1', 'promo_lag_1', 'promo_effect' = 'promotion_flag' * 'lag_1', 'delivery_lag_7', 'stock_lag_1', 'category_rolling_mean_7', 'category_rolling_std_7', 'price_rolling_mean_7', 'promo_rolling_7', 'stockout_flag', 'rolling_stockouts_7', 'time_since_promo', 'time_since_delivery'**
5. Сезонность: **'month_sin', 'month_cos', 'dow_sin', 'dow_cos'**
6. Перекрестный эффект: **'promo_dow' = 'promotion_flag' * 'day_of_week'; 'price_x_stock' = 'price_unit' * 'stock_available'**



Модели

- Наивный прогноз
- MXGBoost / LightGBM
- Random Forest
- LSTM Sequence model



Наивный прогноз

Модель временного ряда, в которой его текущее значение равно предыдущему наблюдаемому значению этого ряда. «Наивная» модель — самый примитивный метод прогнозирования.

Описывается выражением:

$$y(t+1)=y(t),$$

где $y(t)$ — последнее наблюдаемое значение, $y(t+1)$ — прогнозируемое значение.

```
baseline_df1 = df1.dropna(subset=['lag_1', 'units_sold']).copy()  
baseline_df1['pred_baseline'] = baseline_df1['lag_1']
```



Baseline Model (Naive Forecast):

MAE: 10.86

RMSE: 15.24

R^2 : -0.674

Отрицательный R^2 говорит о том, что выбранная модель не подходит для данных. Она не может уловить закономерности и предсказывает значения хуже, чем просто использование среднего значения целевой переменной. Поэтому применю для предсказания более сложную модель.



LightGBM

LightGBM была описана Голинь К., и соавт. в статье 2017 года под названием «LightGBM: A Highly Efficient Gradient Boosting Decision Tree».

- LightGBM

Столбцы, которые не будут использованы в качестве признаков для модели:

```
exclude_cols = ['date', 'sku', 'brand', 'segment', 'channel', 'region', 'pack_type', 'delivered_qty', 'units_sold', 'category']
```

Столбцы, которые будут использованы в качестве признаков для модели:

```
features = [col for col in df1.columns if col not in exclude_cols and not np.issubdtype(df1[col].dtype, np.datetime64)]
```

Из них являются категориальными:

```
categorical_features = ['sku_cat', 'segment_cat', 'category_cat', 'day_of_week', 'month', 'is_weekend', 'promotion_flag', 'is_start_of_month', 'is_end_of_month']
```

Целевая переменная:

```
target = 'units_sold'
```



LightGBM

- LightGBM

```
def train_and_evaluate_lgbm(X_train, X_test, y_train, y_test):  
    model = lgb.LGBMRegressor(n_estimators=100, learning_rate=0.05, max_depth=5, random_state=42,  
verbose=-1)  
    model.fit(X_train, y_train)  
    y_pred = model.predict(X_test)
```



LightGBM Regressor Results:

MAPE: 0.22

SMAPE: 0.23

MAE: 3.54

RMSE: 5.00

R²: 0.774



LightGBM

- LightGBM дополненная лагами и скользящими переменными:

```
for lag in lag_days: df1_lgbm[f'units_sold_lag_{lag}'] = (df1_lgbm.groupby(['channel', 'region'])['units_sold'].shift(lag))
for window in rolling_windows: df1_lgbm[f'rolling_mean_{window}'] = (
    df1_lgbm.groupby(['channel', 'region'])['units_sold']
    .shift(1).rolling(window=window).mean().reset_index(level=0, drop=True))
df1_lgbm[f'rolling_std_{window}'] = (df1_lgbm.groupby(['channel', 'region'])['units_sold']
    .shift(1).rolling(window=window).std().reset_index(level=0, drop=True))
```



LightGBM Regressor Results:

MAPE: 0.22

SMAPE: 0.23

MAE: 3.57

RMSE: 5.03

R^2 : 0.771

- Улучшения метрик не произошло



LightGBM

- LightGBM Extended Log-Transformed Target:

```
def train_lgbm_log_target(X_train, X_test, y_train,  
y_test):
```

```
    y_train_log = np.log1p(y_train)  
    y_test_log = np.log1p(y_test)  
    model = LGBMRegressor(random_state=42)  
    model.fit(X_train, y_train_log)  
    y_pred_log = model.predict(X_test)  
    y_pred = np.expm1(y_pred_log)
```



LightGBM with Log1p Target:

MAPE: 0.21

SMAPE 0.23

MAE: 3.38

RMSE: 4.85

R²: 0.788



LightGBM

- LightGBM+avg_by_channel_region

```
df1['avg_by_channel_region'] = (df1.groupby(['channel', 'region', 'date'])['units_sold'].transform('mean'))
```

LightGBM Regressor Results:

MAPE: 0.20

SMAPE: 0.22

MAE: 2.78

RMSE: 3.99

R²: 0.751

- Модель показала наилучшую среди LightGBM производительность



Random Forest

- Random Forest

```
rf_model = RandomForestRegressor(n_estimators=100,  
                                random_state=42)  
rf_model.fit(X_train_rf, y_train_rf)  
y_pred_rf = rf_model.predict(X_test_rf)
```



Random Forest with Log1p Target:

MAPE: 0.5
SMAPE 0.39
MAE: 5.71
RMSE: 6.86
R²: 0.263

- Random Forest GridSearchCV

```
y_train_rf_log = np.log1p(y_train_rf)  
param_grid = {  
    'n_estimators': [100, 200],  
    'max_depth': [5, 10, None],  
    'min_samples_split': [2, 5],  
    'min_samples_leaf': [1, 2],  
    'max_features': ['sqrt', 0.8]  
}  
rf = RandomForestRegressor(random_state=42, n_jobs=-1)  
grid_search = GridSearchCV(  
    estimator=rf,  
    param_grid=param_grid,  
    scoring='neg_mean_absolute_error',  
    cv=3,  
    verbose=1,  
    n_jobs=-1  
)  
grid_search.fit(X_train_rf, y_train_rf_log)  
grid_search.best_params_, grid_search.best_score_best_rf_log =  
    grid_search.best_estimator_  
y_pred_best_rf_log = best_rf_log.predict(X_test_rf)  
y_pred_best_rf = np.expm1(y_pred_best_rf_log)
```



Best Random Forest (GS):

MAPE: 0.44
SMAPE 0.33
MAE: 4.99
RMSE: 6.22
R²: 0.393



LSTM Sequence model

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.LSTM(32, activation='relu', input_shape=(X_train_lstm.shape[1], X_train_lstm.shape[2])))
model.add(tf.keras.layers.Dense(1, activation='relu'))
model.compile(optimizer='adam', loss='mean_squared_error')
model.fit(X_train_lstm, y_train_lstm, epochs=100, batch_size=32,
        callbacks=[EarlyStopping(patience=10, restore_best_weights=True)],
        verbose=1)
```



LSTM Results:

MAE: 7.25

RMSE: 10.24

R^2 : 0.053

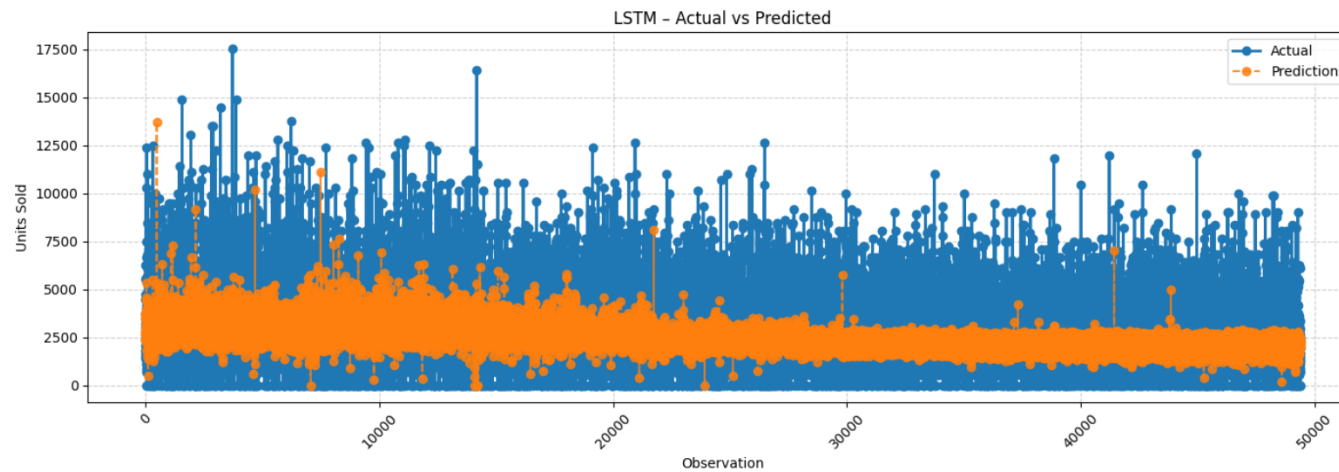
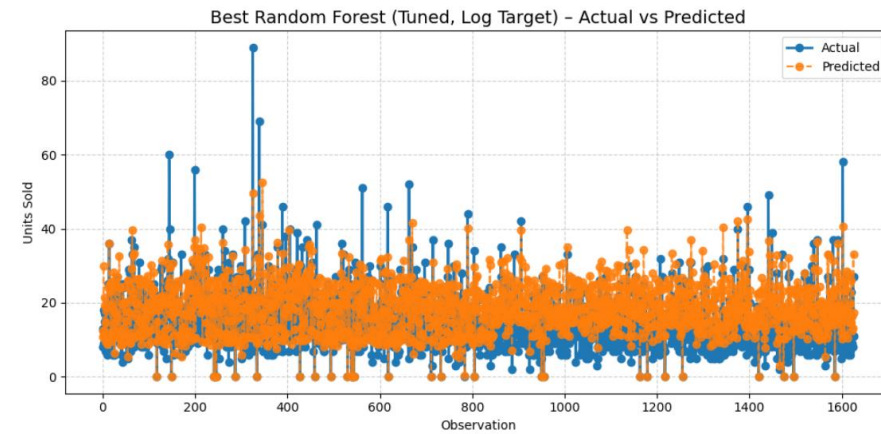
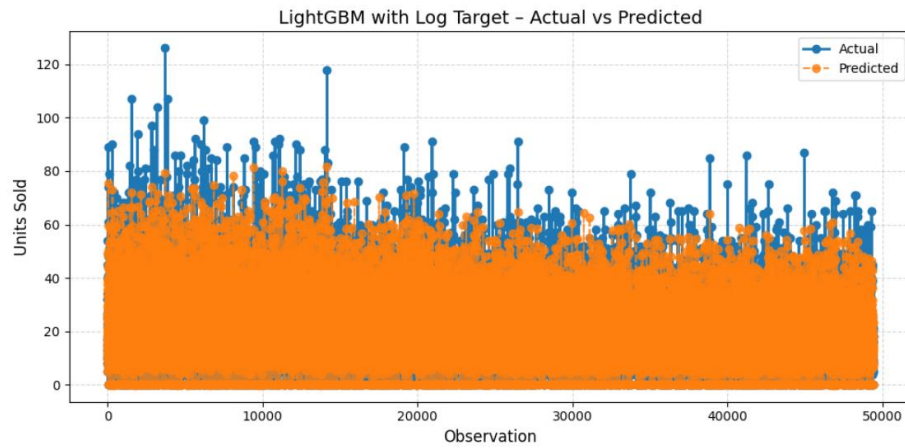


Результаты исследования

Вариант модели	MAE	RMSE	R^2
Наивный прогноз	10.86	15.24	-0.674
LightGBM Regressor	3.54	5.00	0.774
LightGBM Extended	3.57	5.03	0.771
LightGBM Extended Log-Transformed Target	3.38	4.85	0.788
LightGBM+avg_by_cha nnel_region	2.78	3.99	0.751
Random Forest	5.71	6.86	0.263
RF GridSearchCV	4.99	6.22	0.393
LSTM Sequence model	7.38	10.35	0.032



Результаты исследования





Выводы

- Предсказанные моделями значения в разной степени отражают тенденции продаж
- Лучшую производительность среди примененных показала модель вида LightGBM
- Лучший результат среди моделей LightGBM - у модели с дополнительным параметром `avg_by_channel_region`
- Худший результат – у наивного прогноза и LSTM
- Чтобы исключить отрицательные значения в прогнозных данных модели LSTM необходимо использовать функцию активации ReLU (Rectified Linear Unit) в последнем слое модели



Продолжение исследования

- Изменить глубину прогноза до еженедельных и ежеквартальных продаж.
- Для улучшения качества прогноза LSTM более детально подбирать параметры модели
- Применить гибридные модели, которые основаны на усредненных предсказаниях ранее используемых мной моделей.
- Использовать возможности библиотеки Prophet для прогнозирования (Sean J. Taylor, Benjamin Letham *"Forecasting at scale"*)
- Выполнить прескриптивный анализ, используя машинное обучение с подкреплением (RL), для автоматизации процесса принятия решений на основе созданных прогнозов

