

Финансовый прогноз с использованием библиотеки statsmodels и Prophet*Анишкова Анастасия Сергеевна**Приамурский государственный университет имени Шолом-Алейхема**Студент***Аннотация**

Цель исследования - разработать и проанализировать финансовый прогноз временных рядов с использованием библиотек Python statsmodels (модель SARIMAX) и Prophet, чтобы оценить динамику цен на акции компании Apple и повысить точность прогнозирования за счёт учёта тренда, сезонности и внешних факторов. Для реализации использовалась облачная платформа для создания и выполнения кода на Python Google Colab. Полученный результат можно использовать как учебное пособие.

Ключевые слова: Google Colab, SARIMAX, Prophet, statsmodels, Python.

Financial forecast using the statsmodels and Prophet libraries*Anishkova Anastasia Sergeevna**Sholom-Aleichem Priamurskiy State University**Student***Abstract**

The purpose of the study is to develop and analyze a financial forecast of time series using the Python libraries statsmodels (SARIMAX model) and Prophet to evaluate the dynamics of Apple's stock prices and improve the accuracy of forecasting by taking into account the trend, seasonality, and external factors. The study used the Google Colab cloud platform for creating and executing Python code. The resulting data can be used as a tutorial.

Keywords: Google Colab, SARIMAX, Prophet, statsmodels, Python.

1 Введение**1.1 Актуальность**

Анализ и прогнозирование финансовых временных рядов играют ключевую роль в принятии обоснованных инвестиционных решений, управлении рисками и планировании экономической деятельности. В условиях высокой волатильности финансовых рынков возникает необходимость использования эффективных методов прогнозирования, способных учитывать такие особенности временных рядов, как тренд, сезонность и влияние внешних факторов.

1.2 Обзор исследований

В. А. Селезнева, Г. Ю. Чаплиев описали применение python в экономическом и статистическом анализе [1]. Обзор open-source библиотек для решения задач прогнозирования временных рядов провели Е. А. Свекольников, В. Н. Пановский [2]. В. В. Альчаков, В. А. Крамарь провели оценку методов машинного обучения для прогнозирования сезонных временных рядов [3]. Применимости статистических и машинных моделей для прогнозирования инфляции оценили И. А. Астраханцева, А. С. Герасимов, О. П. Смирнова [4]. Л. С. Цветков порекомендовал системы для логистических компаний [5]

1.3 Цель исследования

Цель исследования — разработать и проанализировать финансовый прогноз временных рядов с использованием библиотек Python statsmodels (модель SARIMAX) и Prophet, чтобы оценить динамику цен на акции компании Apple и повысить точность прогнозирования за счёт учёта тренда, сезонности и внешних факторов.

2 Материалы и методы

Для проведения исследования использовалась среда Google Colab, язык программирования Python и библиотеки Prophet и statsmodels.

3 Результаты

Ссылка на ноутбук:
<https://colab.research.google.com/drive/1n0aPRlMlI7ItJwDZX7q1fxgOLp6kDiTk?usp=sharing>.

Подготовим аналитическую среду для работы с временными рядами: установим библиотеку ufinance для загрузки рыночных данных с Yahoo Finance, Prophet от Meta для прогнозирования с учётом сезонности и пропусков, а также statsmodels с её мощной моделью SARIMAX, позволяющей учитывать тренды и сезонные колебания (рис.1).

Для удобства обработки данных подключим pandas и numpy, а для визуализации результатов — matplotlib. Вместе эти инструменты создают идеальную экосистему для глубокого анализа и точного прогнозирования финансовых временных рядов.

```

# Установка необходимых библиотек
!pip install yfinance --upgrade
!pip install prophet --upgrade
!pip install statsmodels

import pandas as pd
import numpy as np
import yfinance as yf
import matplotlib.pyplot as plt

from statsmodels.tsa.statespace.sarimax import SARIMAX
from prophet import Prophet

Requirement already satisfied: multitasking>=0.0.7 in /usr/local/lib/python3.11/dist-packages (from yfinance) (0.0.11)
Requirement already satisfied: platformdirs>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from yfinance) (4.3.8)
Requirement already satisfied: pytz>=2022.5 in /usr/local/lib/python3.11/dist-packages (from yfinance) (2025.2)
Requirement already satisfied: frozendict>=2.3.4 in /usr/local/lib/python3.11/dist-packages (from yfinance) (2.4.6)
Requirement already satisfied: peewee>=3.16.2 in /usr/local/lib/python3.11/dist-packages (from yfinance) (3.18.1)
Requirement already satisfied: beautifulsoup4>=4.11.1 in /usr/local/lib/python3.11/dist-packages (from yfinance) (4.13.4)
Requirement already satisfied: curl_cffi>=0.7 in /usr/local/lib/python3.11/dist-packages (from yfinance) (0.11.3)
Requirement already satisfied: protobuf>=3.19.0 in /usr/local/lib/python3.11/dist-packages (from yfinance) (5.29.5)
Requirement already satisfied: websockets>=13.0 in /usr/local/lib/python3.11/dist-packages (from yfinance) (15.0.1)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.11/dist-packages (from beautifulsoup4>=4.11.1->yfinance) (2.7)
Requirement already satisfied: typing-extensions>=4.0.0 in /usr/local/lib/python3.11/dist-packages (from beautifulsoup4>=4.11.1->yfinance) (4.14.0)
Requirement already satisfied: cffi>=1.12.0 in /usr/local/lib/python3.11/dist-packages (from curl_cffi>=0.7->yfinance) (1.17.1)
Requirement already satisfied: certifi>=2024.2.2 in /usr/local/lib/python3.11/dist-packages (from curl_cffi>=0.7->yfinance) (2025.4.26)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas>=1.3.0->yfinance) (2.9.0.post0)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas>=1.3.0->yfinance) (2025.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests>=2.31->yfinance) (3.4.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests>=2.31->yfinance) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests>=2.31->yfinance) (2.4.0)
Requirement already satisfied: pycparser in /usr/local/lib/python3.11/dist-packages (from cffi>=1.12.0->curl_cffi>=0.7->yfinance) (2.22)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas>=1.3.0->yfinance) (1.17.0)
Downloading yfinance-0.2.63-py2.py3-none-any.whl (118 kB)
118.4/118.4 kB 2.1 MB/s eta 0:00:00

Installing collected packages: yfinance
Attempting uninstall: yfinance
  Found existing installation: yfinance 0.2.62
  Uninstalling yfinance-0.2.62:
    Successfully uninstalled yfinance-0.2.62
Successfully installed yfinance-0.2.63
Requirement already satisfied: prophet in /usr/local/lib/python3.11/dist-packages (1.1.7)
Requirement already satisfied: cmdstanpy>=1.0.4 in /usr/local/lib/python3.11/dist-packages (from prophet) (1.2.5)
Requirement already satisfied: numpy>=1.40.0 in /usr/local/lib/python3.11/dist-packages (from prophet) (2.0.2)

```

Рисунок 1. Установка и импортирование необходимых библиотек

Загружаем исторические данные о цене акций Apple за 9 лет, преобразуем их в удобный формат с учётом рабочих дней и выбираем из них временной ряд цен закрытия, который можно использовать в задачах анализа и прогнозирования временных рядов, например, с помощью моделей SARIMAX или Prophet (рис.2).

```

# Загрузка исторических данных по акциям
ticker = 'AAPL'
data = yf.download(ticker, start='2015-01-01', end='2024-01-01')
data = data.asfreq('B') # бизнес-дни

# Используем закрытие цены (Close) как целевую переменную
ts = data['Close']
print(ts.head())

<ipython-input-2-587003720>:3: FutureWarning: YF.download() has changed argument auto_adjust default to True
data = yf.download(ticker, start='2015-01-01', end='2024-01-01')
[*****100%*****] 1 of 1 completed Ticker AAPL
Date
2015-01-02 24.288576
2015-01-05 23.604332
2015-01-06 23.606552
2015-01-07 23.937565
2015-01-08 24.857307

```

Рисунок 2. Загрузка исторических данных

Обучим и визуализируем прогноз (рис.3) с использованием модели SARIMAX — одной из мощных статистических моделей для анализа и прогнозирования временных рядов с учётом сезонности и тренда. Сначала создаётся модель SARIMAX с заданными параметрами, где `order=(1,1,1)` определяет параметры ARIMA-модели (авторегрессия, разность, скользящее

среднее), а `seasonal\_order=(1,1,1,5)` учитывает еженедельную сезонность (пять рабочих дней в неделю). Затем модель обучается на временном ряде цен закрытия акций Apple с помощью метода `fit()`, при этом отключён вывод логов с помощью параметра `disp=False`. После обучения модели делается прогноз на 30 дней вперёд с помощью метода `get\_forecast()`, а также рассчитываются доверительные интервалы для прогноза. Далее строится график, на котором отображаются последние 100 дней исторических данных, полученный прогноз на месяц вперёд и зона доверительного интервала вокруг прогноза, что позволяет визуальнo оценить динамику и неопределённость предсказания.

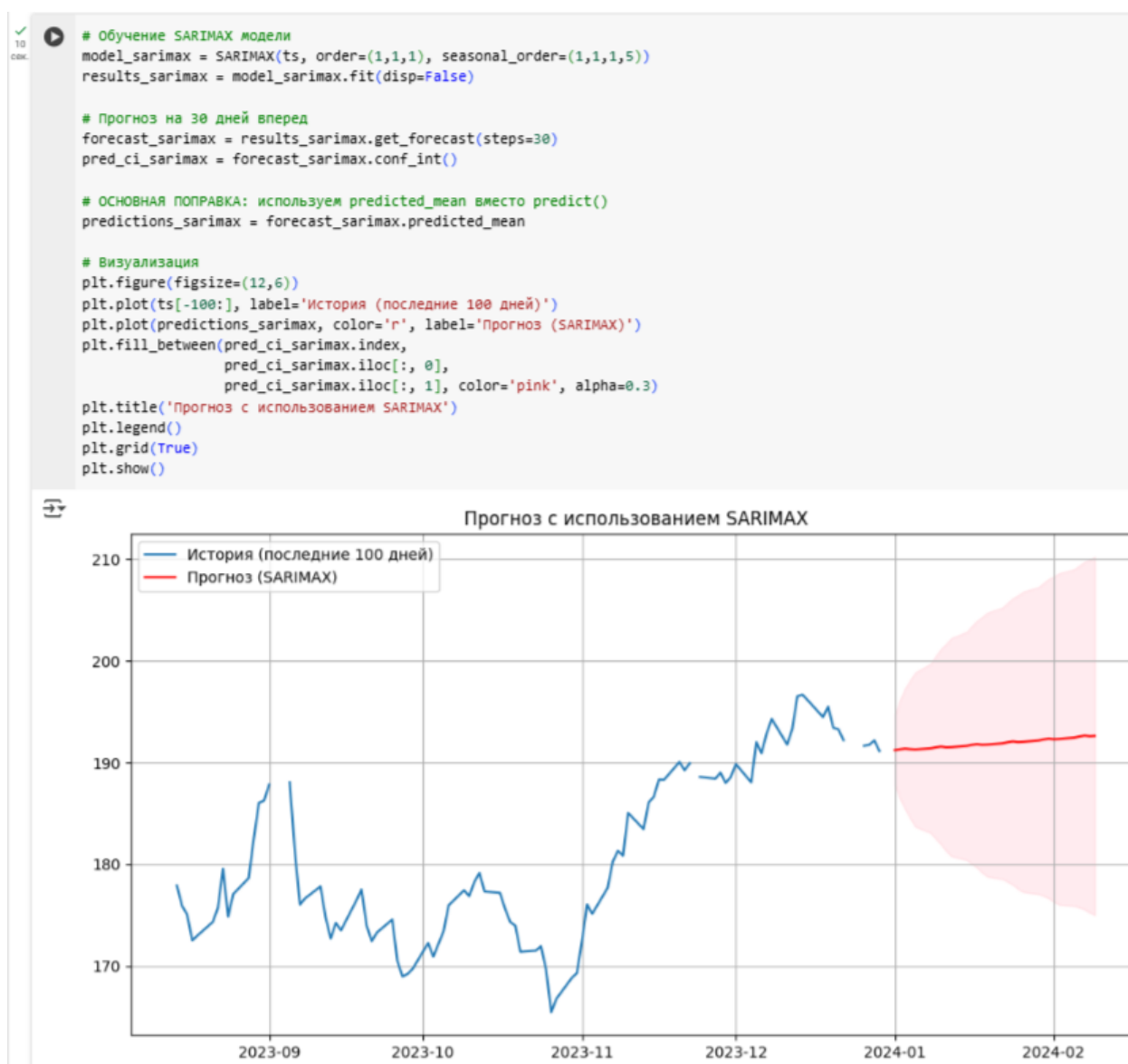


Рисунок 3. Обучение и визуализация прогноза

Подготовим данные и обучим модель Prophet для прогнозирования временного ряда цены закрытия акций Apple (рис.3). Сначала данные преобразуются в формат, подходящий для Prophet: индекс даты, переносится в отдельный столбец, а столбцы переименовываются в 'ds' (дата) и 'y' (значение

ряда). Затем создаётся объект модели Prophet, в который добавляются праздничные дни США с помощью метода `add_country_holidays`, чтобы учесть их влияние на динамику временного ряда. Модель обучается на подготовленных данных с помощью метода `fit()`. Далее с помощью `make_future_dataframe` сгенерируем будущие даты на 30 дней вперёд, после чего выполним прогноз с использованием метода `predict()`. Результат прогноза отображается на графиках (рис. 4,5,6): первый показывает общий прогноз в сравнении с историческими данными, второй — компоненты временного ряда, такие как тренд, сезонность и влияние праздников.

```
✓ 5 сек. # Подготовка данных для Prophet
df_prophet = ts.reset_index()
df_prophet.columns = ['ds', 'y']

# Обучение модели Prophet
model_prophet = Prophet(daily_seasonality=False)
model_prophet.add_country_holidays(country_name='US') # добавляем праздники США
model_prophet.fit(df_prophet)

# Создание будущих дат
future = model_prophet.make_future_dataframe(periods=30)
forecast = model_prophet.predict(future)

# График прогноза
fig2 = model_prophet.plot(forecast)
plt.title("Прогноз с использованием Prophet")
plt.grid(True)
plt.show()

# Компоненты тренда, сезонности и т.д.
fig = model_prophet.plot_components(forecast)
```

Рисунок 3. Подготовка данных и обучение модели

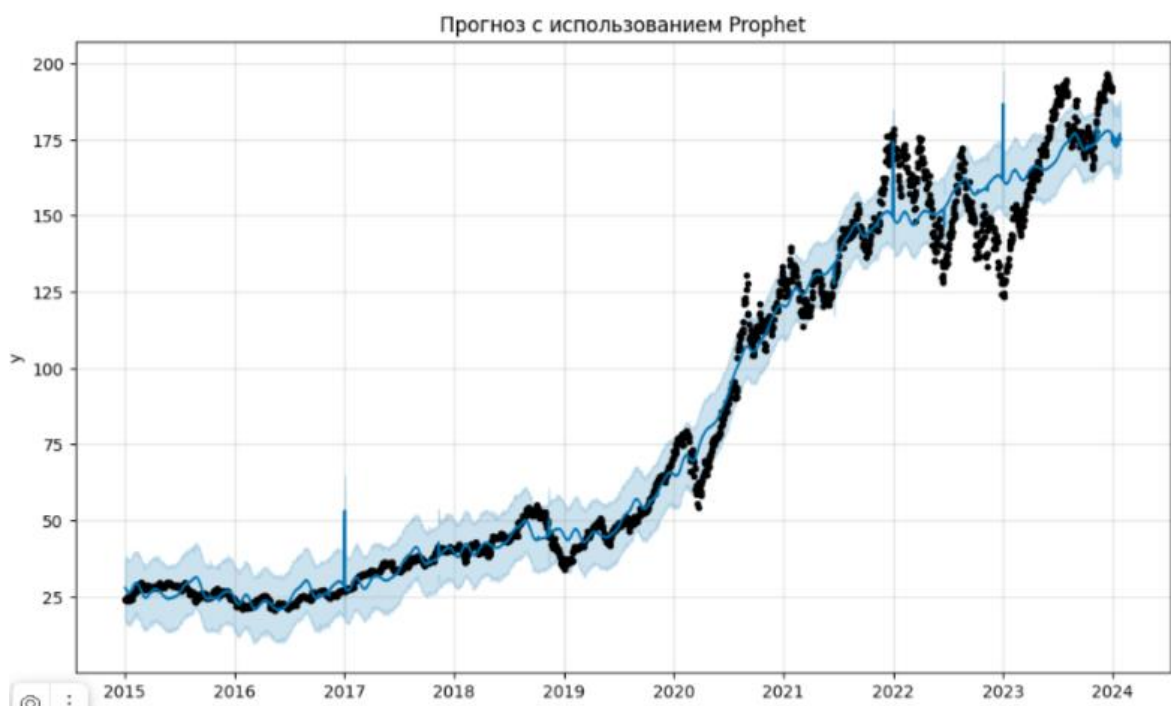


Рисунок 4. Прогноз

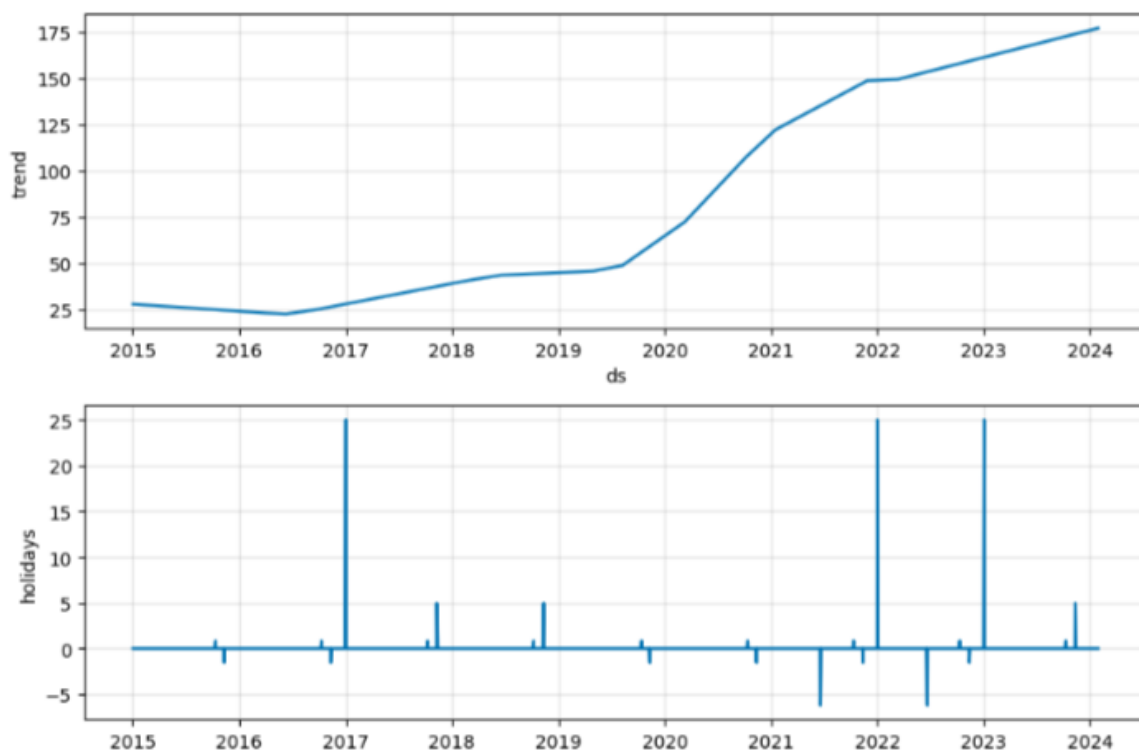


Рисунок 5. Прогноз

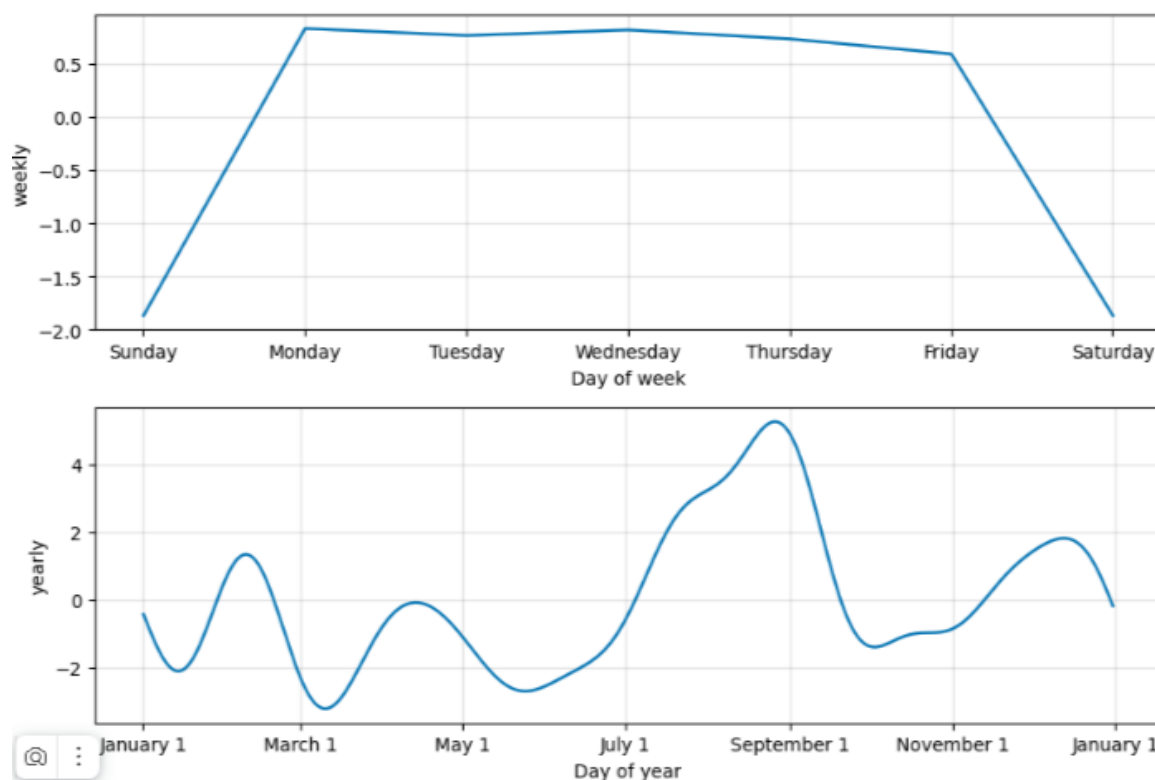


Рисунок 6. Прогноз

Графики компонентов временного ряда, построенные с помощью Prophet, показывают, что данные обладают чётко выраженным восходящим трендом на протяжении всего периода, что указывает на устойчивый рост. Праздники оказывают локальное влияние на отдельные дни, вызывая

краткосрочные колебания. Недельная сезонность демонстрирует небольшие различия между днями недели, с пиками в середине недели и спадом к выходным. Годовая сезонность имеет циклический характер с максимумом в сентябре–октябре и минимумом в январе, что свидетельствует о регулярных ежегодных колебаниях. Общий анализ компонентов позволяет глубже понять структуру данных и повысить точность прогнозирования за счёт учёта всех факторов — тренда, сезонности и внешних событий.

Сравнение методов прогнозирования временного ряда с использованием SARIMAX и Prophet показывает различия в подходах к анализу данных, а также в точности и интерпретируемости результатов. SARIMAX эффективен для базового прогнозирования с явно выраженным трендом и сезонностью, но требует большей настройки. Prophet предлагает более глубокий анализ, автоматически выявляя тренд, сезонность и внешние факторы, что делает его более подходящим для реальных данных с комплексными паттернами.

Библиографический список

1. Селезнева В. А., Чаплиев Г. Ю. Применение python в экономическом и статистическом анализе // Актуальные вопросы общества, науки и образования. 2024. №10. С. 13.
2. Свекольников Е. А., Пановский В. Н. Обзор open-source библиотек для решения задач прогнозирования временных рядов // Моделирование и анализ данных. – 2024. Т. 14. №. 2. С. 45-61.
3. Альчаков В. В., Крамарь В. А. Оценка методов машинного обучения для прогнозирования сезонных временных рядов // Известия Южного федерального университета. Технические науки. 2023. №. 2 (232). С. 250-263.
4. Астраханцева И. А., Герасимов А. С., Смирнова О. П. Оценка применимости статистических и машинных моделей для прогнозирования инфляции // Современные наукоёмкие технологии. Региональное приложение. 2024. Т. 79. №. 3. С. 120-131.
5. Цветков Л. С. Рекомендательные системы для логистических компаний // Journal of Monetary Economics and Management. 2024. №. 7. С. 213-216.