ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

Применение и комбинация статистических методов прогнозирования и технологий машинного обучения для решения задач прогнозирования временных рядов

План (обобщённый вариант)

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

1) Установка временного окна прогнозирования пользователем;
2) Считывание датасета и его базовая предобработка;
3) Проведение базового разведочного анализа временных рядов, содержащихся в датасете;
4) Формирование функции вызова и настройка параметров модели класса «deep learning»;
5) Проведение процедуры прогнозирования временных рядов в датасете;
6) Вывод результатов работы построенных моделей прогнозирования для каждого временного ряда датасета и общего затраченного времени на это.

Формирование окна прогнозирования

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

```
#Установка периода прогнозирования:
from datetime import datetime
start_time = datetime.now()
print('\033[1m' + 'УСТАНОВКА ПЕРИОДА ПРОГНОЗИРОВАНИЯ:' + '\033[0m')
cd1     = 'Начальный месяц для месячного прогнозирования и его дневной детализации (в формате "ГГГГ-ММ"): '
cd2     = 'Конечный месяц для месячного прогнозирования (в формате "ГГГГ-ММ"): '
daily   = input(cd1)
monthly = input(cd2)
inp     = 'salesdaily.csv' # Путь к файлу с историческим фактом
```

УСТАНОВКА ПЕРИОДА ПРОГНОЗИРОВАНИЯ:

Начальный месяц для месячного прогнозирования и его дневной детализации (в формате "ГГГГ-ММ"): 2019-08

Конечный месяц для месячного прогнозирования (в формате "ГГГГ-ММ"): 2020-02

Предобработка данных

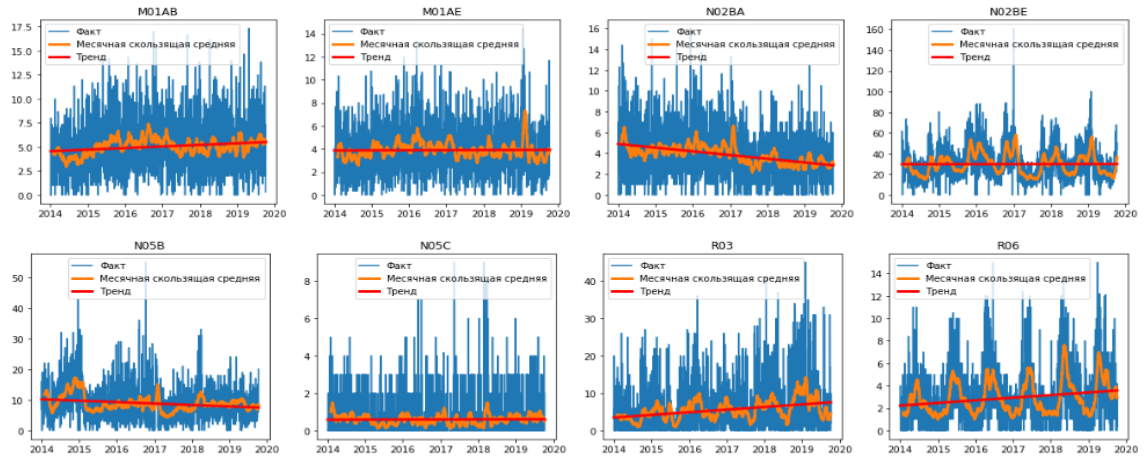
ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

```
dynam = pd.read_csv(inp).dropna(how='all')
dynam['datum'] = pd.to_datetime(dynam['datum'])
dynam = dynam.sort_values(by='datum')
```

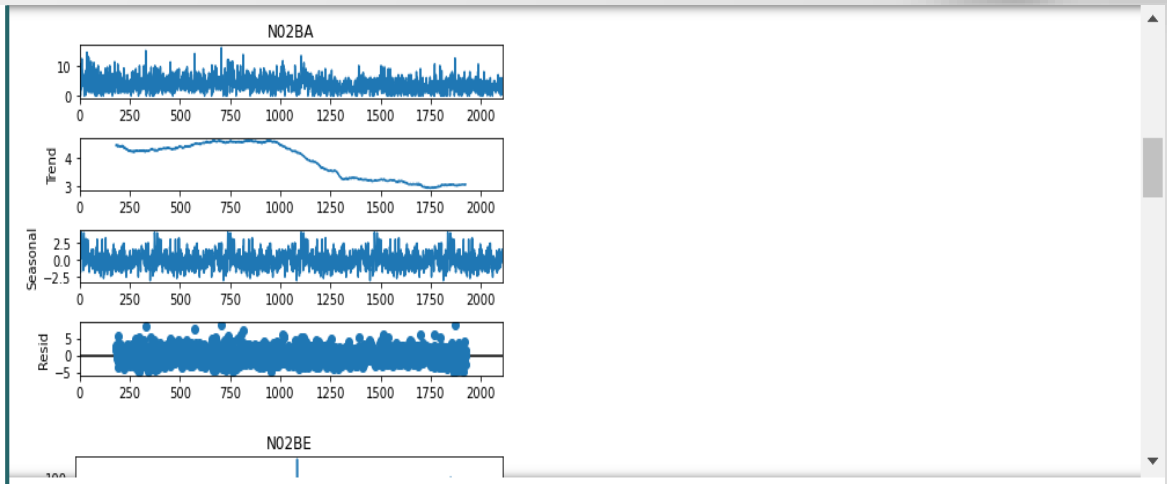
Разведочный анализ

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

ДИНАМИКА ПРОДАЖ РАЗНЫХ КАТЕГОРИЙ ЛЕКАРСТВ:

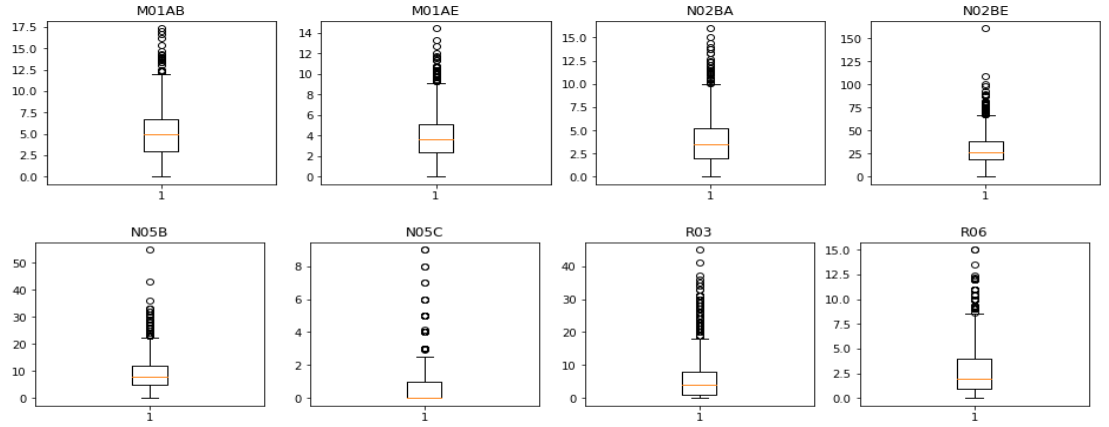


Как можно заметить: все паттерны фактов очень разные, что говорит о том, что, скорее всего, будут подходить разные методы прогнозирования для разных категорий товаров. Есть как восходящие тренды, так и затухающие, а также и практически полное отсутствие направленного движения ряда данных вовсе. Тем не менее, у всех динамик фактов есть сезонность: как аддитивная, так и мультипликативная, что говорит о том, что стоит рассматривать только те методы прогнозирования, которые способны учитывать сезонные колебания.



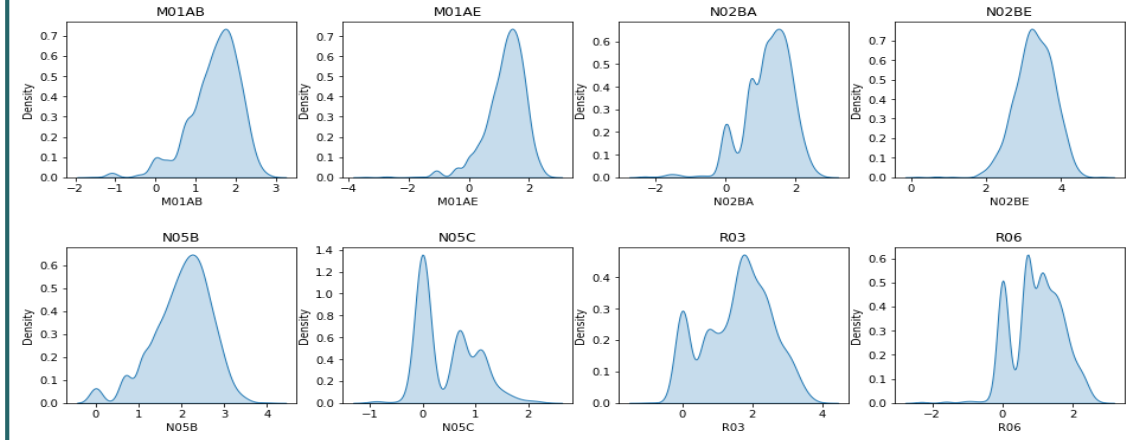
При декомпозиции временных рядов разных категорий лекарств можно прийти к выводу о том, что при из разбитии на тренд-сезонная модель хорошо объясняет их поведение. Колебания остатков происходят в довольно ограниченной амплитуде практически во всех категориях, кроме "NO2BE", что говорит о том, что поведение рассматриваемых временных рядов не совсем хаотично и может быть объяснено характером их компонент.

"ЯЩИКИ С УСАМИ" ПРОДАЖ РАЗНЫХ КАТЕГОРИЙ ЛЕКАРСТВ:



При условии нескольких тысяч точек значений выбросы единичны. Наиболее существенный выброс наблюдается в ранее уже упомянутом "проблемной" категории "N02BE". Тем не менее, датасет сформирвоан из реальныз данных, поэтому никакие записи удалять нельзя. Можно считать, что иногда, по каким-то причинам, бывают аномальные изменения величины спроса на разные категории лекарств, и будущая прогнозная модель должна достойно работать в рамках этого допущения.

РАСПЕРЕДЕЛЕНИЕ ФАКТОВ РАЗНЫХ КАТЕГОРИЙ ЛЕКАРСТВ:



Как можно заметить: все факты категорий лекарств, кроме "N05C", "K03" и "R06", имеют распределение, близкое к нормальному, поэтому работа прогнозных моделей с ними будет упрощена. В категориях "N05C", "K03" и "R06" имеет место быть двупиковость с ассиметрией, которая, скорее всего, была вызвана наличием несколько "аномальных" значений в некоторые дни продаж. Тем не менее, ввиду реальности данных прогнозирование будет проводиться на неизменённом датасете.

Формирование функции и параметров нейронной сети

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

```python
#Формирование параметров нейронной сети:
def get_model(params, input_shape):
        model = Sequential()
        model.add(LSTM(units=params["lstm_units"], return_sequences=True, input_shape=(input_shape, 1)))
        model.add(Dropout(rate=params["dropout"]))

        model.add(LSTM(units=params["lstm_units"], return_sequences=True))
        model.add(Dropout(rate=params["dropout"]))

        model.add(LSTM(units=params["lstm_units"], return_sequences=True))
        model.add(Dropout(rate=params["dropout"]))

        model.add(LSTM(units=params["lstm_units"], return_sequences=False))
        model.add(Dropout(rate=params["dropout"]))

        model.add(Dense(1))

        model.compile(loss=params["loss"],
                      optimizer=params["optimizer"],
                      metrics=[RootMeanSquaredError(), MeanAbsoluteError()])
        return model

params = {
        "loss": "mean_squared_error",
        "optimizer": "adam",
        "dropout": 0.1,
        "lstm_units": 300,
        "epochs": 50,
        "batch_size": 2,
        "es_patience": 10
         }
```

Формирование вектора дневных весов

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

```
for view in col: #Перебор всех категорий лекарств
    count += 1
    #Выделение связанных с целевым месяцем месяцев для расчёта дневной детализации:
    dynamics = dynam[['datum', view]].copy()
    dynamics_target = dynamics[(dynamics['datum'] >= start) & (dynamics['datum'] <= end)]
    dynamics_past = dynamics[(dynamics['datum'] >= (start - pd.DateOffset(years=year_diff))) & (dynamics['datum'] <= (end - pd.Da
    dynamics_previous = dynamics[(dynamics['datum'] >= (start - pd.DateOffset(months=1))) & (dynamics['datum'] <= (end - pd.Date(
    dynamics_prepast = dynamics[(dynamics['datum'] >= (start - pd.DateOffset(years=year_diff) - pd.DateOffset(months=1))) & (dyna
    dynamics_preprevious = dynamics[(dynamics['datum'] >= (start - pd.DateOffset(months=2))) & (dynamics['datum'] <= (end - pd.Da

    #Уравнение дней в месяцах для дневной детализации:
    if  dynamics_past.shape[0] == 30:
        dynamics_previous = dynamics_previous.iloc[0:30,]
        dynamics_prepast = dynamics_prepast.iloc[0:30,]
        dynamics_preprevious = dynamics_preprevious.iloc[0:30,]
    elif dynamics_past.shape[0] == 31:
        if dynamics_previous.shape[0] != 31:
            dynamics_previous = dynamics_previous.append(dynamics_previous.iloc[29:30,0:2])
        if dynamics_preprevious.shape[0] != 31:
            dynamics_preprevious = dynamics_preprevious.append(dynamics_preprevious.iloc[29:30,0:2])
        if dynamics_prepast.shape[0] != 31:
            dynamics_prepast = dynamics_prepast.append(dynamics_prepast.iloc[29:30,0:2])
    elif dynamics_previous.month == 1 and dynamics_previous.year % 4 == 0 and dynamics_previous.year % 100 != 0 or dynamics_previ
        dynamics_previous = dynamics_previous.iloc[0:29,0:2]
        dynamics_prepast = dynamics_prepast.iloc[0:29,0:2]
        dynamics_preprevious = dynamics_preprevious.iloc[0:29,0:2]
    else:
        dynamics_previous = dynamics_previous.iloc[0:28,0:2]
        dynamics_prepast = dynamics_prepast.iloc[0:28,0:2]
        dynamics_preprevious = dynamics_preprevious.iloc[0:28,0:2]

    #Сброс индексов в вырезанных месяцах:
    dynamics_target = dynamics_target.reset_index(drop=True)
    dynamics_past = dynamics_past.reset_index(drop=True)
    dynamics_previous = dynamics_previous.reset_index(drop=True)
    dynamics_prepast = dynamics_prepast.reset_index(drop=True)
    dynamics_preprevious = dynamics_preprevious.reset_index(drop=True)

    #Расчёт оптимальных коэффициентов комбинирования:
    dynamics_previous1 = pd.DataFrame()
    def optimization(x):
        dynamics_previous1[view] = x[0]*dynamics_prepast[view] + x[1]*dynamics_preprevious[view]
        coef = sum(abs(dynamics_previous1[view] - dynamics_previous[view]))
        return(coef)
    res = minimize(optimization, [2,2], method='SLSQP')

    #Расчёт ежедневных коэффициентов для дневной детализации целевого месяца:
    short_forecast = dynamics_past
    short_forecast['datum'] += pd.DateOffset(years=year_diff)
    short_forecast[view] = res['x'][0]*dynamics_past[view] + res['x'][1]*dynamics_previous[view]
    short_forecast[view] = short_forecast[view]/sum(short_forecast[view])
```

№ шага	Действие	Примечание
1	k1*[2019–06] + k2*[2018–07] - [2019–07]	k1 и k2 линейно оптимизируются для минимизации разницы (k1 и k2 могут быть только положительными)
2	k1*[2019–07] + k2*[2018–08] = [2019–08]	применение сохранённых k1 и k2 для "прогнозирования " первого месяца месячного прогноза
3	[каждое значение] в [2019–08] / сумма([2019–08])	получение нормированного вектора с "весами" дней
4	[значение первого месяца месячного прогноза] * [вектор с "весами" дней] (из шага № 3)	распределение месячного прогноза на дни

Ex-post прогнозирование

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

```
#Разбиение датасета на обучающую и тестовую выборки:
difference = (dynamics_previous.iloc[0,0].year - dynamics.iloc[0,0].year) * 12 + dynamics_previous.iloc[0,0].month
cut = difference*0.2 - 1
cut = round(cut, 0)
cut = dynamics_previous.iloc[0,0] - pd.DateOffset(months=cut)
dynamics_train = dynamics[(dynamics['datum'] >= dynamics.iloc[0,0]) & (dynamics['datum'] < cut)]
dynamics_test = dynamics[(dynamics['datum'] >= cut) & (dynamics['datum']<=(end-pd.DateOffset(months=1)))]
dynamics_full = dynamics.groupby(dynamics['datum'].dt.strftime('%Y-%m'))[view].sum()
dynamics_train = dynamics_train.groupby(dynamics_train['datum'].dt.strftime('%Y-%m'))[view].sum()
dynamics_test = dynamics_test.groupby(dynamics_test['datum'].dt.strftime('%Y-%m'))[view].sum()
```

```
LSTM_model.fit(
    xtrain,
    ytrain,
    validation_data=(xtest, ytest),
    epochs=params["epochs"],
    batch_size=params["batch_size"],
    verbose=0,
       )

LSTM_test = LSTM_model.predict(xtest)
LSTM_test = pd.concat([pd.DataFrame(sarima_test.reset_index()), pd.DataFrame(LSTM_test)], axis = 1)[['index', 0]]
LSTM_test['index'] = pd.to_datetime(LSTM_test['index'], errors='coerce')
LSTM_test = LSTM_test.groupby(LSTM_test['index'].dt.strftime('%Y-%m')).sum()
```

```
hw_test_model = ExponentialSmoothing(dynamics_train, seasonal = hw_seas, seasonal_periods=12).fit() #Метод Хольта-Уинтерса
hw_test = hw_test_model.predict(start=cut, end=(start-pd.DateOffset(months=1)))

ETS_test_model = ETSModel(dynamics_train, seasonal = ets_seas, seasonal_periods=12).fit() #ETS-модель
ETS_test = ETS_test_model.predict(start=cut, end=(start-pd.DateOffset(months=1)))

sarima_test_model = SARIMAX(dynamics_train, freq='MS', enforce_stationarity=False, enforce_invertibility=False).fit() #SARIMA
sarima_test = sarima_test_model.predict(start=cut, end=(start-pd.DateOffset(months=1)))
```

```
errors = pd.DataFrame(columns=['Метод', MAE, RMSE, MAPE])
hw_row = {'Метод':'Хольт-Унитерс', MAE: round(mae(dynamics_test, hw_test), 2), RMSE: round(mse(dynamics_test, hw_test)**0.5,
ETS_row = {'Метод':'ETS', MAE: round(mae(dynamics_test, ETS_test), 2), RMSE: round(mse(dynamics_test, ETS_test)**0.5, 2), MA
sarima_row = {'Метод':'SARIMA', MAE: round(mae(dynamics_test, sarima_test), 2), RMSE: round(mse(dynamics_test, sarima_test)*
LSTM_row = {'Метод':'LSTM', MAE: round(mae(dynamics_test, LSTM_test), 2), RMSE: round(mse(dynamics_test, LSTM_test)**0.5, 2)
errors = errors.append(hw_row, ignore_index=True)
errors = errors.append(ETS_row, ignore_index=True)
errors = errors.append(sarima_row, ignore_index=True)
errors = errors.append(LSTM_row, ignore_index=True)
```

СТАТИСТИЧЕСКИЕ ОШИБКИ НА ТЕСТОВЫХ ПЕРИОДАХ:

	Метод	MAE (M01AB)	RMSE (M01AB)	MAPE,% (M01AB)	MAE (M01AE)	RMSE (M01AE)	MAPE,% (M01AE)	MAE (N02BA)	RMSE (N02BA)	MAPE,% (N02BA)	MAE (N02BE)	RMSE (N02BE)	MAPE,% (N02BE)	MAE (N05B)	RMSE (N05B)	MAPE,% (N05B)
0	Хольт-Унитерс	13.78	16.46	8.92	13.87	21.58	10.17	9.25	12.27	10.38	108.00	161.91	10.55	43.73	48.20	16.91
1	ETS	13.78	16.46	8.92	13.87	21.58	10.17	9.25	12.27	10.38	118.30	166.95	12.15	43.73	48.23	16.91
2	SARIMA	21.78	26.90	13.12	14.77	26.17	10.10	19.94	24.08	20.38	477.95	555.65	48.18	72.49	80.08	26.77
3	LSTM	11.71	15.30	7.19	17.29	23.98	13.50	32.40	33.82	36.39	562.82	634.13	57.45	27.04	33.87	10.15

**Проверка оптимальности статистических методов прогнозирования при подборе мультипликативной аддитивной сезонностей проводилась ранее (hw_ets и ets_seas).*

Комбинация методов прогнозирования

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

```
#Расчёт ошибок прогноза для комбинации моделей прогнозирования:
exclude = max(errors[RMSE])
for i in range(0,errors.shape[0]):
    if  errors.iloc[i,errors.columns.get_loc(RMSE)] == exclude:
        ex = i
        break
fit_methods = errors.drop([errors.index[ex]])
fit_methods = fit_methods.reset_index(drop=True)
errors_f = errors.drop([errors.index[ex]])

exclude = max(errors_f[RMSE])
for i in range(0,errors_f.shape[0]):
    if  errors_f.iloc[i,errors_f.columns.get_loc(RMSE)] == exclude:
        ex = i
        break
fit_methods = errors_f.drop([errors_f.index[ex]])
fit_methods = fit_methods.reset_index(drop=True)

first_coef = 1 - fit_methods.iloc[0,fit_methods.columns.get_loc(RMSE)]/sum(fit_methods[RMSE])
second_coef = 1 - fit_methods.iloc[1,fit_methods.columns.get_loc(RMSE)]/sum(fit_methods[RMSE])
sum_coef = first_coef + second_coef
first_coef = first_coef / sum_coef
second_coef = second_coef / sum_coef

if [fit_methods.iloc[0,fit_methods.columns.get_loc("Метод")], fit_methods.iloc[1,fit_methods.columns.get_loc("Метод")]]  == |
    combination_test = first_coef*hw_test + second_coef*ETS_test
    row1, row2 = 0,1
elif [fit_methods.iloc[0,fit_methods.columns.get_loc("Метод")], fit_methods.iloc[1,fit_methods.columns.get_loc("Метод")]]  ==
    combination_test = first_coef*ETS_test + second_coef*sarima_test
    row1, row2 = 1,2
elif [fit_methods.iloc[0,fit_methods.columns.get_loc("Метод")], fit_methods.iloc[1,fit_methods.columns.get_loc("Метод")]]  ==
    combination_test = first_coef*hw_test + second_coef*sarima_test
    row1, row2 = 0,2
elif [fit_methods.iloc[0,fit_methods.columns.get_loc("Метод")], fit_methods.iloc[1,fit_methods.columns.get_loc("Метод")]]  ==
    combination_test = first_coef*pd.DataFrame(hw_test).reset_index()[0] + second_coef*LSTM_test.reset_index()[0]
    row1, row2 = 0,3
elif [fit_methods.iloc[0,fit_methods.columns.get_loc("Метод")], fit_methods.iloc[1,fit_methods.columns.get_loc("Метод")]]  ==
    combination_test = first_coef*pd.DataFrame(ETS_test).reset_index()[0] + second_coef*LSTM_test.reset_index()[0]
    row1, row2 = 1,3
elif [fit_methods.iloc[0,fit_methods.columns.get_loc("Метод")], fit_methods.iloc[1,fit_methods.columns.get_loc("Метод")]]  ==
    combination_test = first_coef*pd.DataFrame(sarima_test).reset_index()['predicted_mean'] + second_coef*LSTM_test.reset_inc
    row1, row2 = 2,3


errors_comb = pd.DataFrame(columns=['Метод', MAE, RMSE, MAPE])
comb_row = {'Метод':'Комбинация 2-х лучших методов', MAE: round(mae(dynamics_test, combination_test), 2), RMSE: round(mse(dyr
errors_comb = errors_comb.append(comb_row, ignore_index=True)
```

СТАТИСТИЧЕСКИЕ ОШИБКИ НА ТЕСТОВЫХ ПЕРИОДАХ:

	Метод	MAE (M01AB)	RMSE (M01AB)	MAPE,% (M01AB)	MAE (M01AE)	RMSE (M01AE)	MAPE,% (M01AE)	MAE (N02BA)	RMSE (N02BA)	MAPE,% (N02BA)
0	Хольт-Унитерс	13.78	16.46	8.92	13.87	21.58	10.17	9.25	12.27	10.38
1	ETS	13.78	16.46	8.92	13.87	21.58	10.17	9.25	12.27	10.38
2	SARIMA	21.78	26.90	13.12	14.77	26.17	10.10	19.94	24.08	20.38
3	LSTM	11.71	15.30	7.19	17.29	23.98	13.50	32.40	33.82	36.39
4	Комбинация 2-х лучших методов	12.44	14.97	7.85	13.87	21.58	10.17	9.25	12.27	10.38

Применение оптимальных методов прогнозирования для каждой категории

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

```
if errors_comb[MAPE].iloc[0] < np.min(errors[MAPE]):
    if (row1 == 0) & (row2 == 1):
        long_forecast = first_coef*hw + second_coef*ETS
        tools.append([view, 'Комбинация Хольта-Уинтерса и ETS-модели'])
    elif (row1 == 1) & (row2 == 2):
        long_forecast = first_coef*ETS + second_coef*sarima
        tools.append([view, 'Комбинация ETS-модели и SARIMA'])
    elif (row1 == 0) & (row2 == 2):
        long_forecast = first_coef*hw + second_coef*sarima
        tools.append([view, 'Комбинация Хольта-Уинтерса и SARIMA'])
    elif (row1 == 0) & (row2 == 3):
        long_forecast = first_coef*hw.reset_index()[0] + second_coef*LSTM_for.reset_index()[0]
        tools.append([view, 'Комбинация Хольта-Уинтерса и LSTM'])
        n=+1
    elif (row1 == 1) & (row2 == 3):
        long_forecast = first_coef*ETS.reset_index()[0] + second_coef*LSTM_for.reset_index()[0]
        tools.append([view, 'Комбинация ETS-модели и LSTM'])
        n=+1
    elif (row1 == 2) & (row2 == 3):
        long_forecast = first_coef*sarima.reset_index()['predicted_mean'] + second_coef*LSTM_for.reset_index()[0]
        tools.append([view, 'Комбинация SARIMA и LSTM'])
        n=+1

else:
    include = min(errors[MAPE])
    for i in range(errors.shape[0]):
        if  errors.iloc[i, errors.columns.get_loc(MAPE)] == include:
            inc = i
            break
    if inc == 0:
        long_forecast = hw
        tools.append([view, 'Хольт-Уинтерс'])

    elif inc == 1:
        long_forecast = ETS
        tools.append([view, 'ETS-модель'])
    elif inc == 2:
        long_forecast = sarima
        tools.append([view, 'SARIMA'])
    else:
        long_forecast = LSTM_for
        tools.append([view, 'LSTM'])
        z += 1
```

ОПТИМАЛЬНЫЕ МОДЕЛИ ПРОГНОЗИРОВАНИЯ ДЛЯ КАЖДОЙ КАТЕГОРИИ:

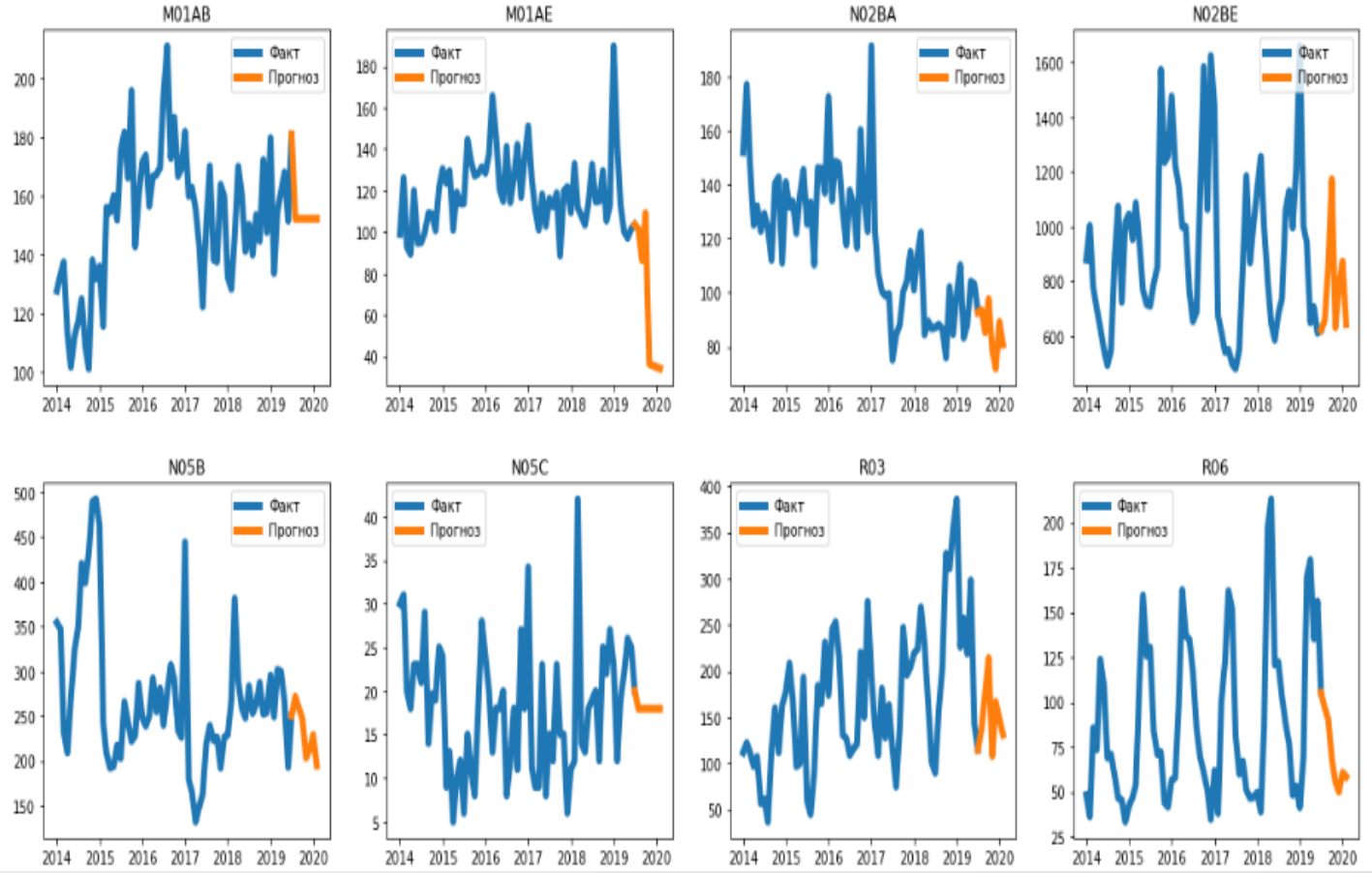
	Категория	Оптимальная модель
0	M01AB	LSTM
1	M01AE	SARIMA
2	N02BA	Хольт-Уинтерс
3	N02BE	Хольт-Уинтерс
4	N05B	Комбинация Хольта-Уинтерса и LSTM
5	N05C	LSTM
6	R03	ETS-модель
7	R06	Хольт-Уинтерс

**Обучение всех методов прогнозирования из доступного набора на полных периодах временных рядов проводилось ранее*

Формирование месячного прогнозного плана

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

ВИЗУАЛИЗАЦИЯ ПРОГНОЗОВ ОПТИМАЛЬНЫХ МОДЕЛЕЙ ДЛЯ КАЖДОЙ КАТЕГОРИИ:



ПРОГНОЗНЫЙ ПЛАН:

	Дата	M01AB	M01AE	N02BA	N02BE	N05B	N05C	R03	R06
0	2019-08	152.2	101.37	93.67	653.71	271.43	17.96	137.49	96.86
1	2019-09	152.2	86.42	85.35	853.80	258.60	17.96	176.51	90.37
2	2019-10	152.2	109.10	97.63	1172.48	246.59	17.96	214.08	68.53
3	2019-11	152.2	36.52	78.60	630.18	203.40	17.96	108.13	55.51
4	2019-12	152.2	35.75	71.99	780.22	212.10	17.96	166.45	50.02
5	2020-01	152.2	35.00	89.17	871.62	228.66	17.96	148.34	60.66
6	2020-02	152.2	34.26	80.91	642.08	194.76	17.96	131.18	58.39

Декомпозиция первого месяца месячного прогнозного плана

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

ДЕТАЛИЗАЦИЯ ПЕРВОГО МЕСЯЦА ПРОГНОЗНОГО ПЛАНА:

	Дата	M01AB	M01AE	N02BA	N02BE	N05B	N05C	R03	R06
0	2019-08-01	2.38	2.01	1.04	13.50	8.74	1.01	7.82	5.56
1	2019-08-02	1.82	2.00	1.70	12.04	10.90	0.00	1.30	4.44
2	2019-08-03	3.98	2.59	2.93	24.57	9.70	0.00	0.96	1.90
3	2019-08-04	5.07	2.46	1.23	10.10	9.94	2.93	2.87	6.98
4	2019-08-05	5.42	2.03	4.77	15.88	7.65	2.03	1.04	1.44
5	2019-08-06	6.99	3.40	2.50	18.06	5.16	0.00	2.00	2.19
6	2019-08-07	3.38	4.50	6.19	19.24	5.16	1.01	3.04	3.45
7	2019-08-08	4.84	3.45	2.08	15.08	12.19	0.12	3.82	2.51
8	2019-08-09	2.65	2.68	4.02	21.48	13.53	0.46	2.87	1.87
9	2019-08-10	7.24	4.62	4.58	18.61	7.27	0.00	0.96	2.51
10	2019-08-11	3.76	4.18	5.39	27.09	12.94	1.01	0.87	7.06
11	2019-08-12	10.92	4.72	5.41	24.02	7.04	0.00	2.08	1.87
12	2019-08-13	3.84	3.85	1.65	19.73	11.39	0.35	0.52	2.84
13	2019-08-14	2.82	3.94	4.77	16.82	6.83	0.00	5.47	3.37
14	2019-08-15	7.96	2.59	3.16	27.14	10.67	1.91	0.00	2.87
15	2019-08-16	4.48	5.31	3.73	20.03	6.94	3.04	2.78	4.34
16	2019-08-17	2.85	1.30	2.54	23.61	6.34	0.90	0.87	1.26
17	2019-08-18	7.79	3.93	0.55	27.09	8.25	1.01	3.91	0.97
18	2019-08-19	6.61	4.65	2.32	23.06	8.96	0.00	3.73	5.92
19	2019-08-20	4.08	4.66	3.00	18.63	6.11	0.00	9.21	2.48
20	2019-08-21	3.61	1.41	1.79	20.40	4.79	1.01	1.13	3.41
21	2019-08-22	3.68	3.17	0.85	15.57	7.17	1.01	0.00	4.38
22	2019-08-23	3.79	2.01	4.67	20.51	10.05	0.00	10.68	2.60
23	2019-08-24	5.26	2.94	1.47	33.49	13.27	0.00	5.56	2.76
24	2019-08-25	4.39	5.33	4.09	22.70	6.69	0.12	11.55	2.17
25	2019-08-26	11.63	4.83	2.50	35.89	11.73	0.00	9.03	3.59
26	2019-08-27	3.32	0.41	5.95	18.29	6.46	0.35	2.08	1.71
27	2019-08-28	2.74	1.80	2.32	19.07	5.63	0.90	0.00	2.02
28	2019-08-29	7.37	3.37	1.65	26.03	11.73	1.01	1.82	2.48
29	2019-08-30	4.61	4.38	3.97	27.58	8.50	0.46	32.22	2.84
30	2019-08-31	2.92	2.86	0.85	18.40	9.71	1.01	7.30	3.08

* ВРЕМЯ ВЫПОЛНЕНИЯ ИТ-РЕШЕНИЯ:
Duration: 0:16:55.158672

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

Применение и комбинация статистических методов прогнозирования и технологий машинного обучения для решения задач прогнозирования временных рядов