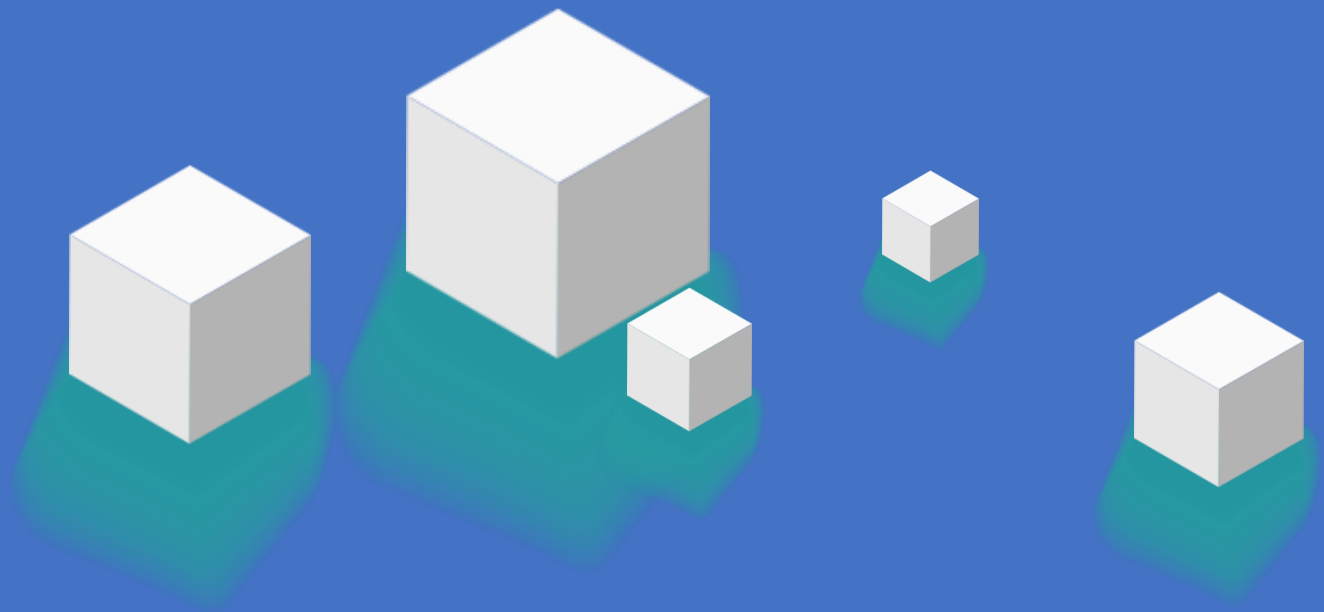
时间序列AI大赛



>> 今天的学习目标

Facebook Prophet

- 时间序列预测工具 prophet
- 饱和增长
- 突变点
- 节日与大事件
- Project A：页面流量预测
- Project B：交通流量预测

时间序列AI大赛

- AI大赛：资金流入流出预测
- 使用prophet进行预测
- 周期因子分析

1/2 Facebook Prophet

Project：沪市指数预测

Project：沪市指数预测

- 沪市指数的历史数据（从1990年12月19日到2020年3月12日）
- 请你编写代码对沪市指数未来3个月（截止到2020年6月30日）的变化进行预测（将数据转化为按月统计即可）

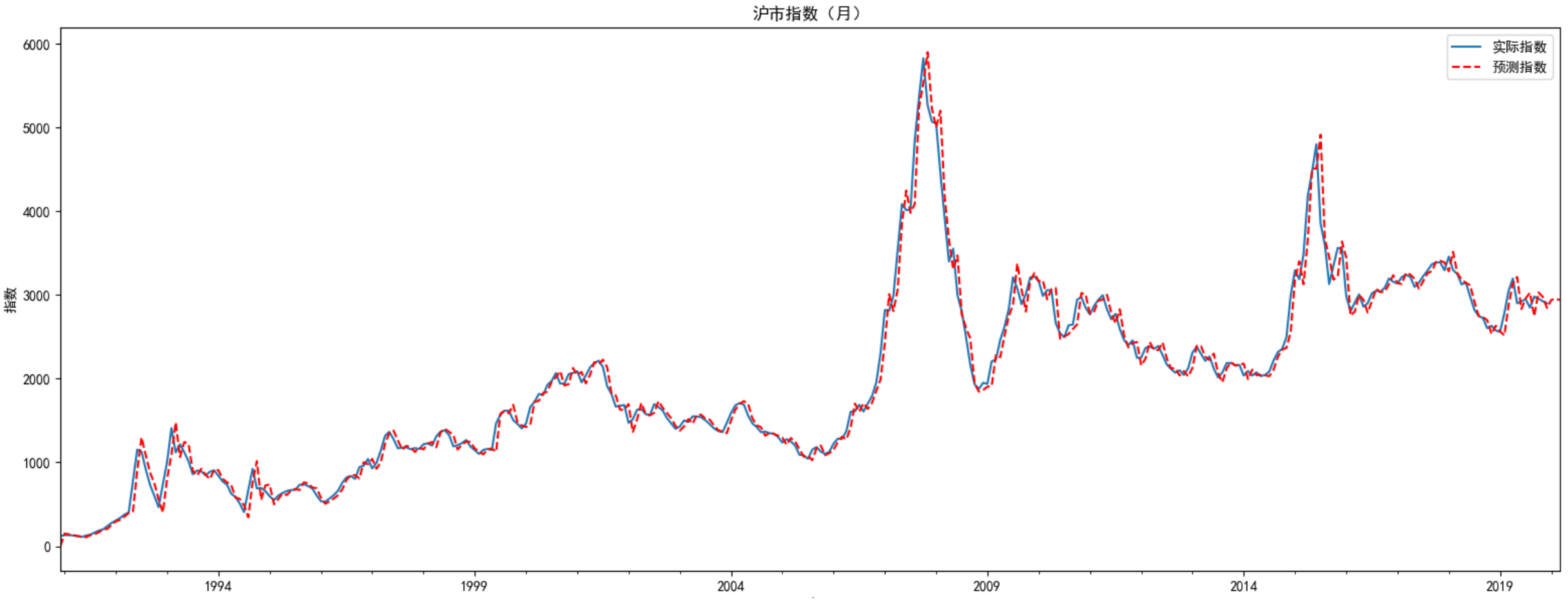
1990年到2020年沪市指数走势

时间序列模型

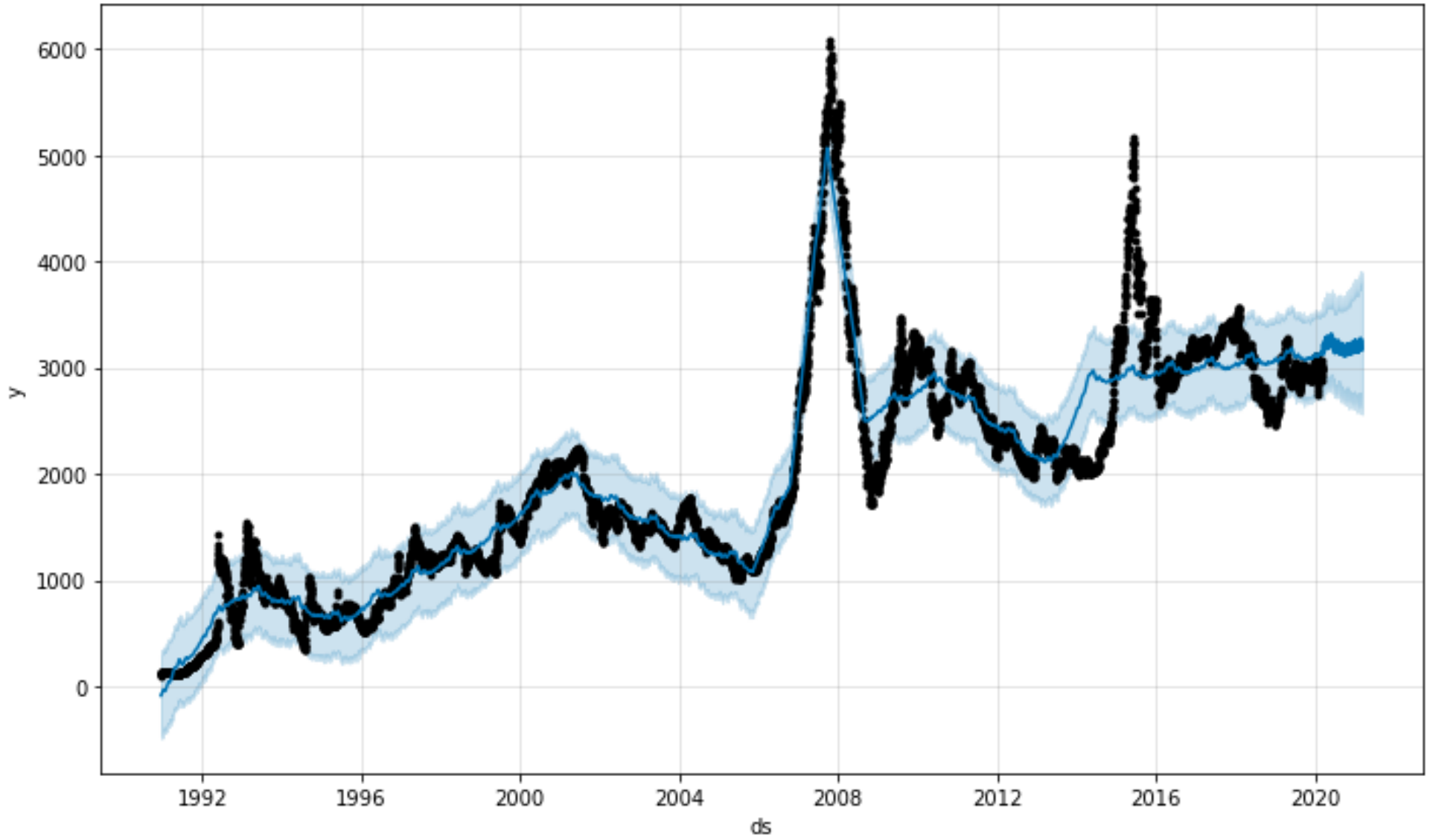
ARMA/ARIMA 统计模型的不足:

- ARMA，要求时序数据是稳定的，现实数据很难符合
- ARIMA，模型为线性模型，无法处理非线性关系，同时要求数据点的间隔等长，比如X1和X2间隔一个小时，那么X2和X3也间隔一个小时
- 如果数据缺失，则需要使用插值等方法来预估缺失值，然后再使用预估值来进行参数拟合，这样会引入噪音

Project：沪市指数预测（ARIMA）



Project：沪市指数预测（Prophet）



Facebook prophet工具

prophet:

- facebook开源的时间序列预测工具
  https://facebook.github.io/prophet/
- Prophet是一个基于相加模型（ additive model）的时间预测，可以精准的拟合非线性的周期趋势
- 对yearly、weekly和daily的周期性使用非线性拟合，亮点在于Prophet模型还添加了holidays（影响因子），可以很好的对节日（比如十一、春节等）带来的活跃数据的突变进行预测

prophet的优势：

- 处理数据丢失问题
- 趋势迁移问题（shifts in the trend）
- 异常的数据点（outliers）

prophet模型： y(t)=g(t)+s(t)+h(t)+e

g(t)代表趋势项，用来表示时间序列中非周期性的变化

s(t)代表周期项，用来表示时间序列中的周期变化

h(t)代表活动效果项，用来表达时间序列中的一些异常活动，例节假日，购物节等

e 用来表示不能被模型所描述的异常误差

Facebook prophet工具

prophet模型： y(t)=g(t)+s(t)+h(t)+e

- g(t)：趋势项在Prophet中有两种实现方法，第一种是饱和增长模型，第二种是分段线性模型
- s(t)：运用傅里叶级数作为周期项，使得预测模型具有灵活的周期效应
- h(t)：实际上，很多节日不是在一年中固定的一天发生，

比如母亲节（五月的第二个礼拜天），中国的春节

在Prophet模型中，用户在进行预测前可以向模型输入活动表格

=> Prophet的预测会更加准确

prophet 工具使用：

- pip3 install prophet
- 遵循 sklearn 库的使用接口，

模型拟合，fit （一般1-5秒）

```
model = Prophet()

model.fit(df)
```

模型预测，predict

```
forecast = model.predict(future)
```

Facebook prophet工具

prophet模型：

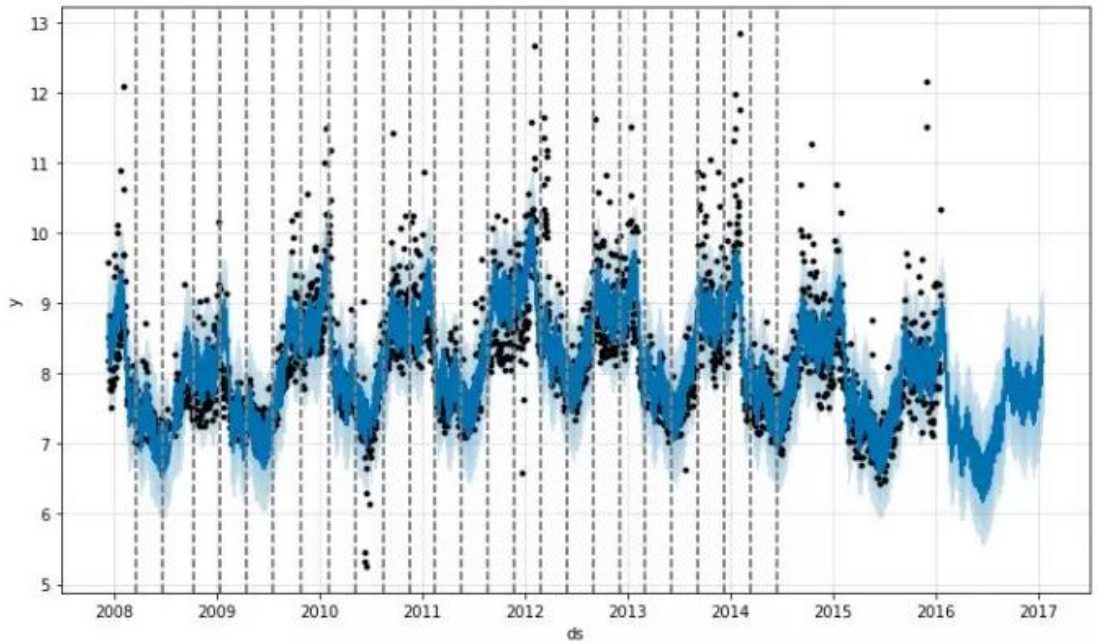
- Trend趋势，对时间序列中的趋势部分拟合分段线性函数，线性拟合会将特殊点和缺失数据的影响降到最小
- 饱和增长

通常情况下，增长会有最大容量限制，比如未来12个月某app在某地区的下载量，最大下载量要小于等于该地区手机用户总数

基于这样的领域知识，分析师可以定义模型的容量限制为C(t)

- 突变点，随着突变点数量的增多，拟合变得更灵活。在研究趋势成分时，分析师要面临两个基本问题，即过拟合与欠拟合

changepoint_prior_scale参数，可以调整趋势的灵活性，解决过拟合/欠拟合，参数值越大，拟合的时间序列曲线越灵活



参数	描述
growth	‘linear’或‘logistic’用来规定线性或逻辑曲线趋势
changepoints	包括潜在突变点的日期列表（不指明则默认为自动识别）
n_changepoints	若不指定突变点，则需要提供自动识别的突变点数量
changepoint_prior_scale	设定自动突变点选择的灵活性

Facebook prophet的突变点分析

Thinking：prohpeth是如何进行突变点分析的？

1）检测突变点

Prophet 默认在时间序列的前 80% 范围内均匀放置 25 个候选突变点。

通过 稀疏先验（Laplace 分布） 对突变点的幅度 δ_s 进行正则化，筛选出显著的点。

2) 分段趋势建模

趋势公式（以线性趋势为例）：

$$g(t) = \left(k + \sum_{s \in S} \delta_s \cdot \mathbb{I}(t \geq s)\right) \cdot t + \left(m + \sum_{s \in S} \gamma_s \cdot \mathbb{I}(t \geq s)\right)$$

δs：突变点 s 处的斜率变化量。

γ：保证函数连续的偏移量（避免突变点处的跳跃）。

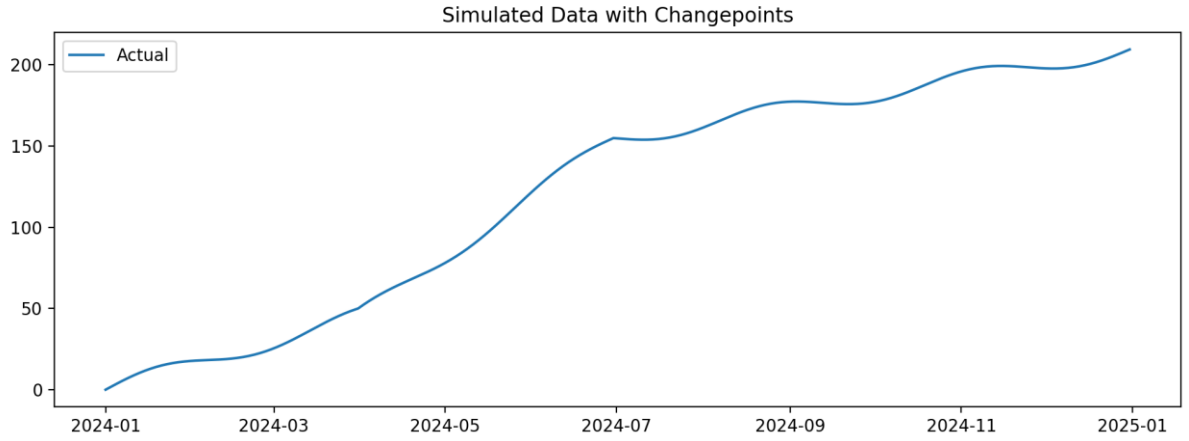
3) 预测未来趋势

未来时间段的趋势延续最后一个突变点后的斜率（即假设未来趋势不再突变）。

Facebook prophet的突变点分析

假设某产品的日销量在以下时间点发生突变：

- 2024-01-01：初始增长（斜率=0.5）。
- 2024-04-01：因促销活动，斜率突增至 1.2。
- 2024-07-01：市场竞争加剧，斜率降至 0.3。




```
# 生成日期范围
dates = pd.date_range(start='2024-01-01', end='2024-12-31')
n_days = len(dates)

# 模拟趋势突变
np.random.seed(42)
trend = np.concatenate([
    0.5 * np.arange(90),                # 1月-3月：斜率0.5
    0.5 * 90 + 1.2 * np.arange(91),     # 4月-6月：斜率1.2
    0.5 * 90 + 1.2 * 91 + 0.3 * np.arange(n_days-90-91) # 7月-12月：斜率0.3
])
```

```
# 添加季节性噪声
y = trend + 5 * np.sin(np.linspace(0, 10*np.pi, n_days))
df = pd.DataFrame({'ds': dates, 'y': y})

# 可视化原始数据
plt.figure(figsize=(12, 4))
plt.plot(df['ds'], df['y'], label='Actual')
plt.title("Simulated Data with Changepoints")
plt.legend()
plt.show()
```

突变点检测.py

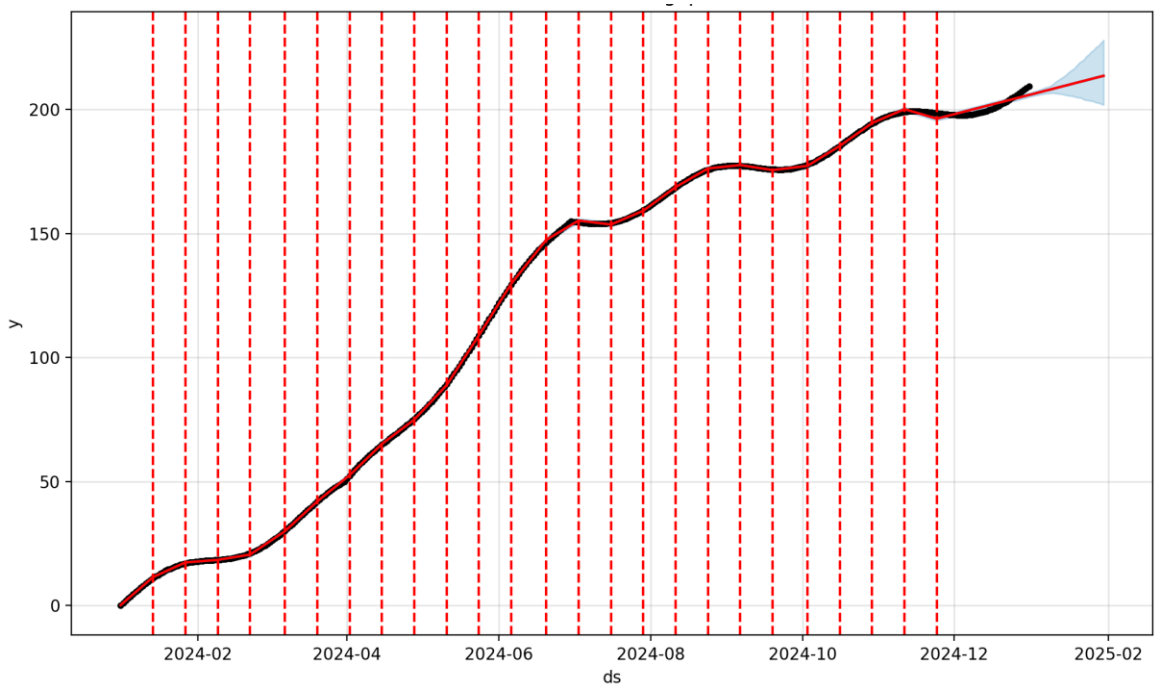
Facebook prophet的突变点分析

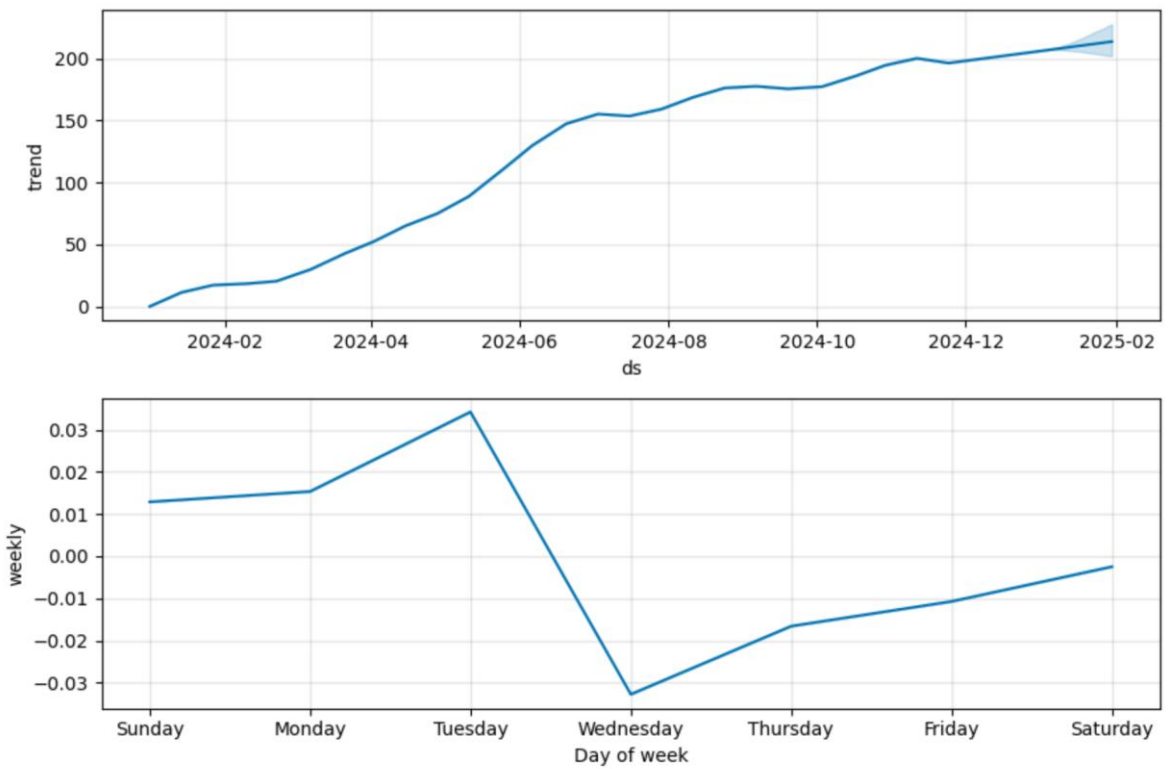
```
## 训练prophet模型
# 初始化模型（显式启用周季节性）
model = Prophet(
    yearly_seasonality=False,
    weekly_seasonality=True,
    changepoint_prior_scale=0.5,  # 提高突变点灵敏度
    changepoint_range=0.9       # 在前90%数据中检测突变点
)

# 拟合数据
model.fit(df)

# 创建未来30天的预测
future = model.make_future_dataframe(periods=30)
forecast = model.predict(future)
```

```
## 可视化突变点与预测
# 绘制预测结果
fig1 = model.plot(forecast)
plt.title("Forecast with Changepoints")

# 标记突变点
from prophet.plot import add_changepoints_to_plot
add_changepoints_to_plot(fig1.gca(), model, forecast)

# 分解趋势和季节性
fig2 = model.plot_components(forecast)
```

突变点检测.py

Facebook prophet的突变点分析





Prophet 的突变点机制能自动适应趋势变化，非常适合业务场景中突发事件的建模。

传统ARIMA模型，假设趋势是平滑变化的，无法灵活处理突发性趋势转折。

Prophet的解决方案：通过突变点检测，Prophet 将时间序列分段建模，每段的趋势斜率可以独立调整。例如：产品销量因营销活动突然增长。或者经济指标因政策调整骤降。

突变点检测.py

Facebook prophet工具

prophet模型：

- 季节性，拟合并预测季节的效果，基于傅里叶级数提出了一个灵活的模型：

$$s(t)=\sum_{n=1}^{N}\left(a_n\cos\left(\frac{2\pi nt}{P}\right)+b_n\sin\left(\frac{2\pi nt}{P}\right)\right)$$

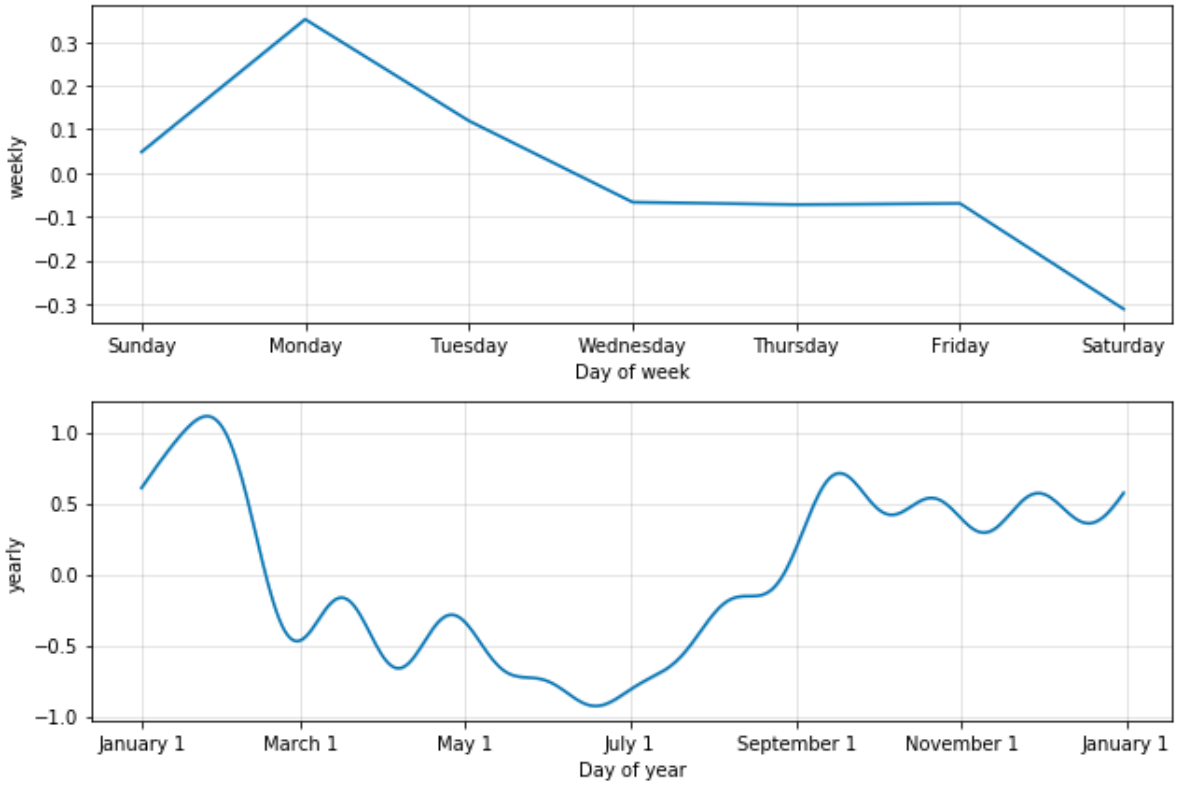
P代表周期，年度的P=365.25，周数据的P=7

对季节性建模，在给定N的情况下，估计参数[a1,b1……aN, bN]

傅里叶阶数N是重要的参数，用来定义模型中是否考虑高频变化：

如果分析师认为高频变化的成分只是噪声，可以将N取较低值

如果不是噪音，可以将N设置为较高值，提升预测精度



Facebook prophet工具

prophet模型：

- 活动效果项，即节假日和大事件

他们都是重要的时间因素，比如中国传统春节，在这个期间人们会购买大量新商品

允许分析师使用过去和未来事件的自定义列表，这些大事件前后的日期将会被单独考虑，并且通过拟合附加的参数模拟节假日和事件的效果

参数	描述
yearly_seasonality	周期为年的季节性
weekly_seasonality	周期为周的季节性
daily_seasonality	周期为日的季节性
holidays	内置的节假日名称和日期
seasonality_priori_scale	改变季节模型的强度
holiday_prior_scale	改变假日模型的强度

季节和假日相关的参数

Project A：页面访问流量预测

Project : 佩顿·曼宁维基百科访问流量预测

- 数据集，维基百科上面对美国橄榄球运动员佩顿·曼宁的日访问记录，2905条数据（2007年12月10日到2016年1月20日）
- Prophet 的输入量通常包含两列的数据框：ds 和 y

ds 列包含日期（YYYY-MM-DD）或者是具体的时间点（YYYY-MM-DD HH:MM:SS）

y 列是数值变量，表示我们希望去预测的量

ds	y
2007/12/10	9.590761139
2007/12/11	8.519590316
2007/12/12	8.183676583
2007/12/13	8.072467369
2007/12/14	7.893572074
2007/12/15	7.783640596
2007/12/16	8.414052432
……	……
2016/1/17	9.273878393
2016/1/18	10.33377535
2016/1/19	9.125871215
2016/1/20	8.891374009

Project A：页面访问流量预测

Prophet工具使用：

- make_future_dataframe方法，将未来的日期扩展指定的天数，得到一个数据框。默认情况下，做会自动包含历史数据的日期，因此也可以用来查看模型对于历史数据的拟合效果
- predict 方法，对每一行future 日期得到预测值（yhat ）预测 forecast 创建的对象应当是新的DataFrame，其中包含一列预测值 yhat ，以及成分的分析和置信区间
- plot_components方法，查看预测的成分分析
- 查看forecast都有哪些列：print(forecast.columns)

成分分析的绘制：

- trend趋势，来自trend字段
- yearly趋势，来自yearly字段
- weekly趋势，来自weekly字段

因为是加法模型，所以：

```
forecast['additive_terms'] = forecast['weekly'] + forecast['yearly']
```

```
forecast['yhat']  = forecast['trend'] +  forecast['additive_terms']
```

```
forecast['yhat']  = forecast['trend'] +forecast['weekly'] +
forecast['yearly']
```

如果有节假日因素，那么

```
forecast['yhat']  = forecast['trend'] +forecast['weekly'] +
forecast['yearly'] + forecast['holidays']
```

Project A：页面访问流量预测

- 查看forecast.tail()

会发现'multiplicative_terms', 'multiplicative_terms_lower', 'multiplicative_terms_upper'这3列为空，因为是加法模型

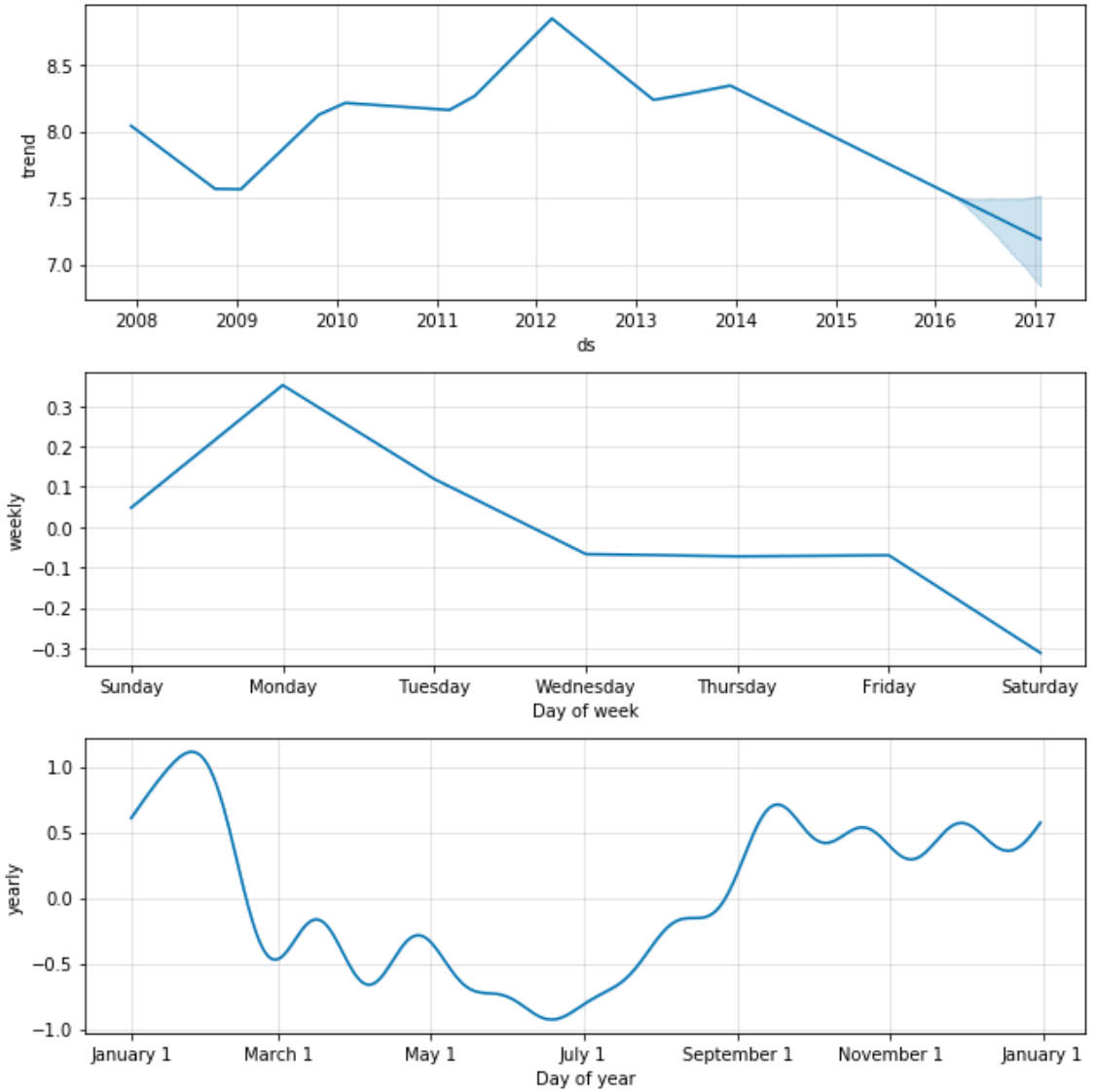
- 成分分析趋势解读

weekly中的Monday为0.035的意思就是，在trend的基础上，加0.035

Saturday为-0.3的意思就是，在trend的基础上，减0.3

因此，weekly这条线的高低反应了销量的趋势

Thinking：一年这种哪个月份，销量最高？



Project A：页面访问流量预测

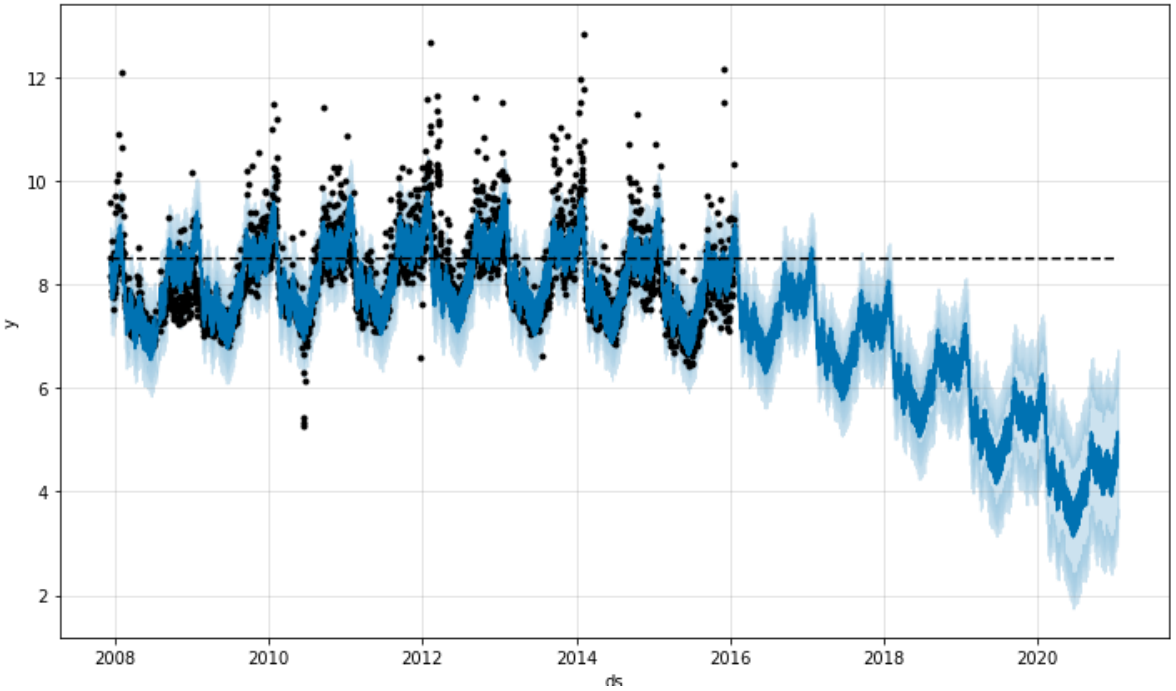
- 预测饱和增长

Prophet在预测增长情况时，会存达到极值，比如总人口数等，这里称为承载能力（carrying capacity），这时上限就是趋于饱和

新建一列 cap 来指定承载能力的大小，通常情况下这个值应当通过市场规模的数据或专业知识来决定，比如

df['cap'] = 8.5

注意：DataFrame每行都必须指定 cap值，但不一定是恒定值，如果市场规模在不断地增长，那么 cap 也可以是不断增长的序列



- 预测饱和减少（市场的最低floor）

logistic增长模型还可以处理饱和最小值，方法与指定最大值的列的方式相同

Project A：页面访问流量预测

- 趋势突变点

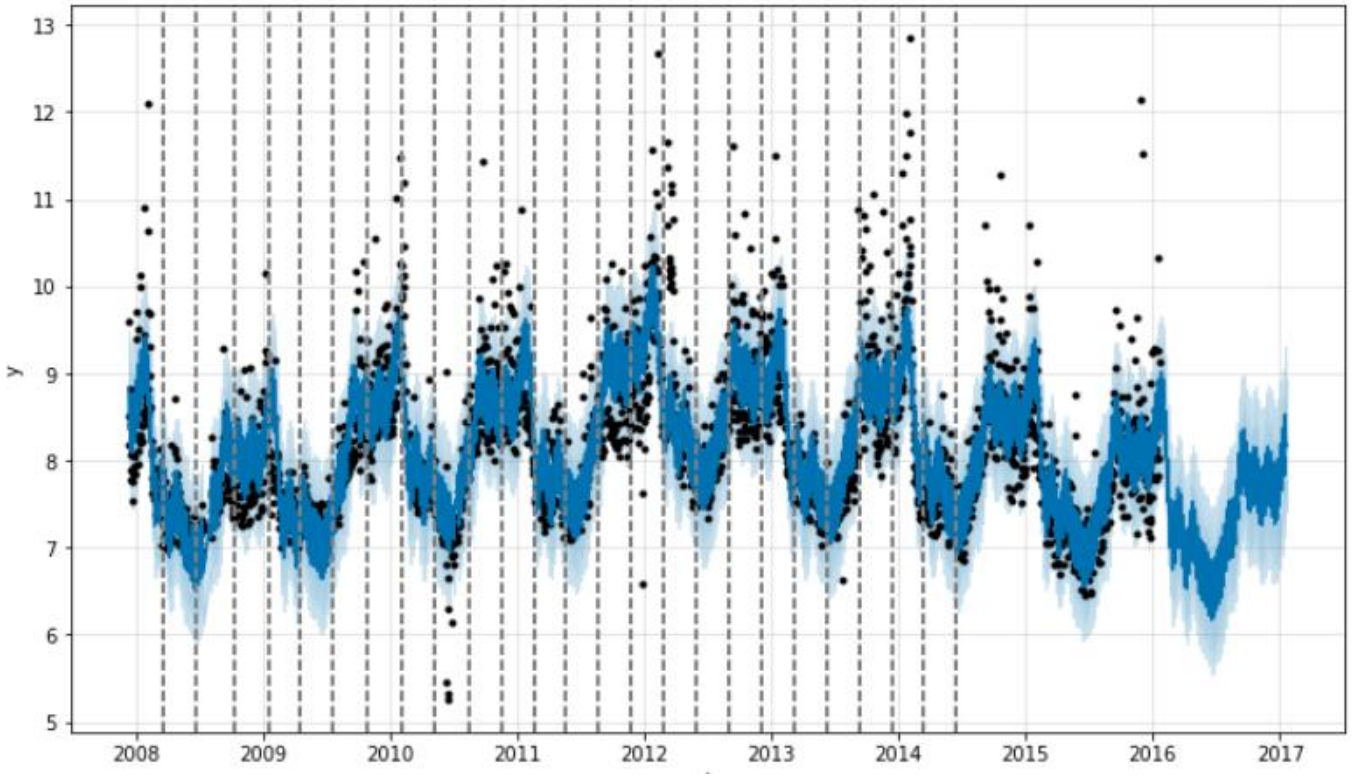
真实的时间序列数据往往存在一些突变点

Prophet 将自动监测到这些点，并对趋势做适当地调整

默认下， Prophet 会识别出 25 个潜在的突变点（均匀分布在在前 80% 的时间序列数据中），绝大多数突变点并不会包含在建模过程中

显示突变点的位置

```
from fbprophet.plot import add_changepoints_to_plot

fig = m.plot(forecast)

a = add_changepoints_to_plot(fig.gca(), m, forecast)
```



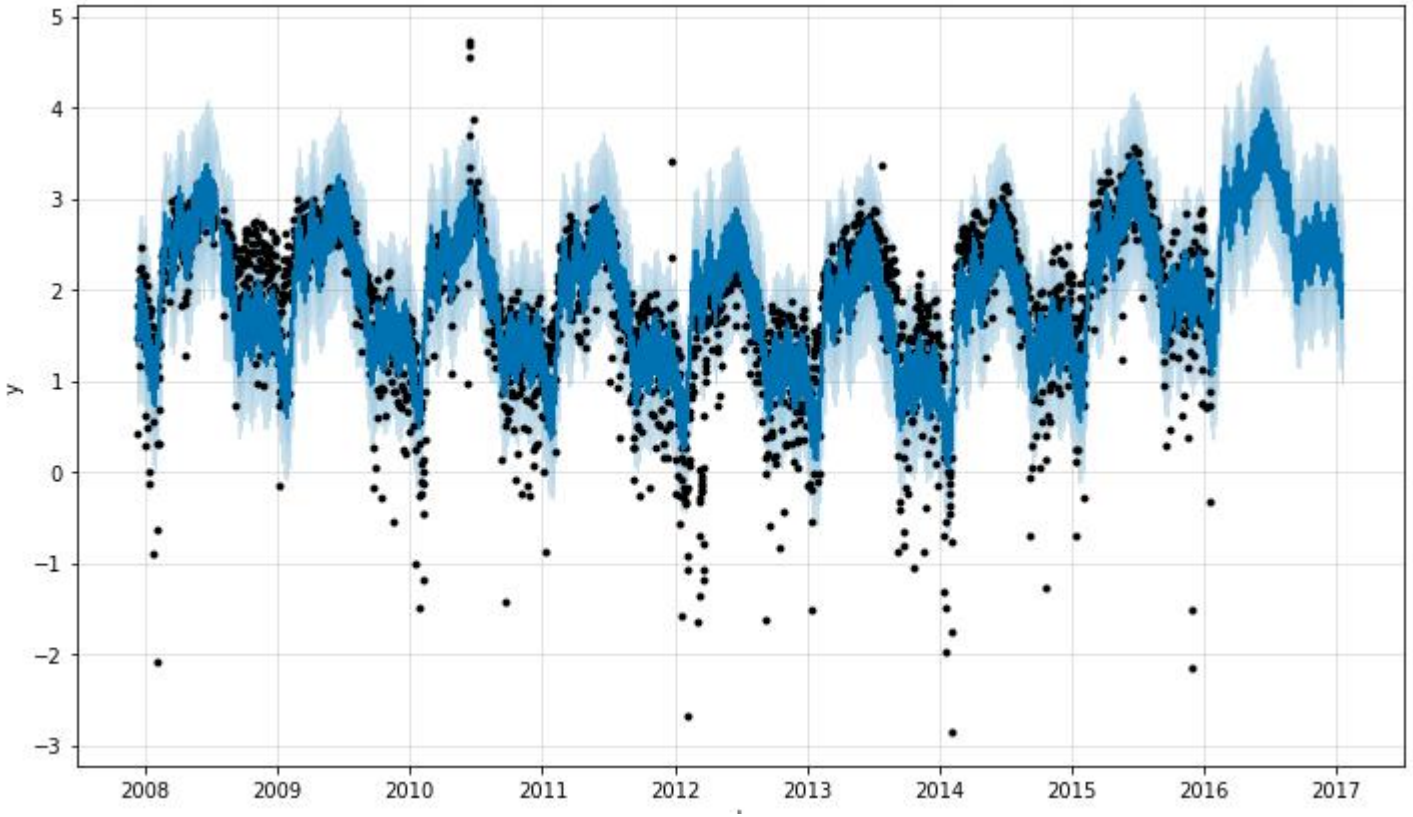
竖线指出这些潜在的突变点所在的位置

Project A：页面访问流量预测

- 指定突变点的位置

使用 changepoints 参数

```
m = Prophet(changepoints=['2014-01-01'])

m.fit(df)

future = m.make_future_dataframe(periods=365)

forecast = m.predict(future)

m.plot(forecast)
```



Project A：页面访问流量预测

- 对节假日建模

创建一个新的DataFrame，包含两列（节假日 holiday 和日期戳 ds ）

注意：这个DataFrame必须包含所有出现的节假日（不仅是历史数据集中，还是要预测的时期中的）

比如，所有佩顿·曼宁参加过的季后赛 与 决赛日期

```
playoffs = pd.DataFrame({
  'holiday': 'playoff',
  'ds': pd.to_datetime(['2008-01-13', '2009-01-03', '2010-01-16',
                        '2010-01-24', '2010-02-07', '2011-01-08',
                        '2013-01-12', '2014-01-12', '2014-01-19',
                        '2014-02-02', '2015-01-11', '2016-01-17',
                        '2016-01-24', '2016-02-07']),
  'lower_window': 0,
  'upper_window': 1,
})
superbowls = pd.DataFrame({
  'holiday': 'superbowl',
  'ds': pd.to_datetime(['2010-02-07', '2014-02-02', '2016-02-07']),
  'lower_window': 0,
  'upper_window': 1,
})
holidays = pd.concat((playoffs, superbowls))
```

Project A：页面访问流量预测

- 对节假日建模

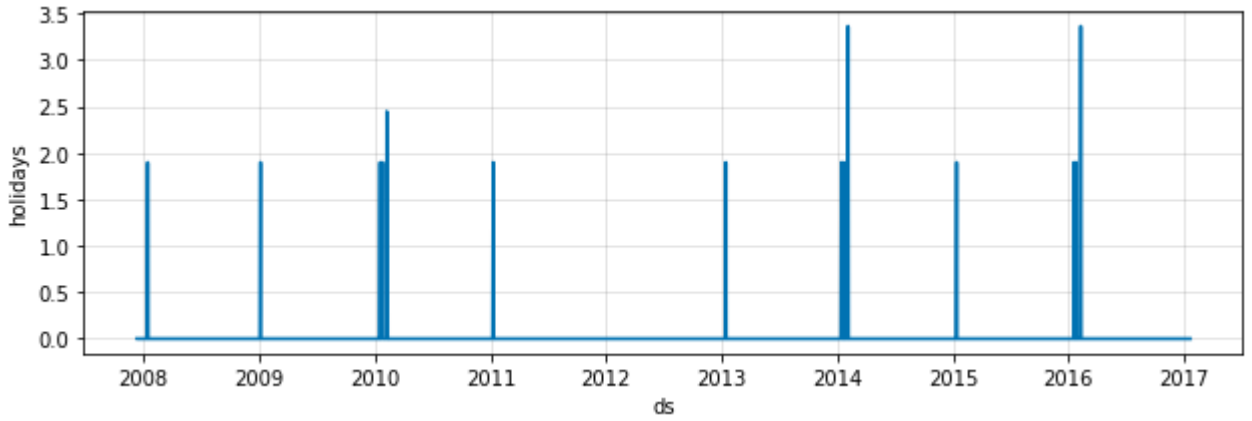
这个DataFrame创建好了，就可以通过传入 holidays

```
m = Prophet(holidays=holidays)

m.fit(df)

future = m.make_future_dataframe(periods=365)

forecast = m.predict(future)
```

可以通过 forecast 数据框，展示节假日效应

```
print(forecast[(forecast['playoff'] +
forecast['superbowl']).abs() > 0][['ds', 'playoff',
'superbowl']][-10:])
```

	ds	playoff	superbowl
2190	2014-02-02	1.217571	1.230312
2191	2014-02-03	1.898042	1.466063
2532	2015-01-11	1.217571	0.000000
2533	2015-01-12	1.898042	0.000000
2901	2016-01-17	1.217571	0.000000
2902	2016-01-18	1.898042	0.000000
2908	2016-01-24	1.217571	0.000000
2909	2016-01-25	1.898042	0.000000
2922	2016-02-07	1.217571	1.230312
2923	2016-02-08	1.898042	1.466063



Project B：交通流量预测

Project B：交通流量预测

- JetRail高铁的乘客数量预测
- 数据集：jetrail.csv，根据过往两年的数据（2012 年 8 月至 2014 年 9月），需要用这些数据预测接下来 7 个月的乘客数量
- 以每天为单位聚合数据集

```
m = Prophet(yearly_seasonality=True, seasonality_prior_scale=0.1)

# 预测未来7个月，213天

future = m.make_future_dataframe(periods=213)
```

ID	Datetime	Count
0	25-08-2012 00:00	8
1	25-08-2012 01:00	2
2	25-08-2012 02:00	6
3	25-08-2012 03:00	2
4	25-08-2012 04:00	2
5	25-08-2012 05:00	2
……	……	……
18286	25-09-2014 22:00	580
18287	25-09-2014 23:00	534

Project：沪市指数预测

Project ： 沪市指数预测

- 沪市指数的历史数据（从1990年12月19日到2020年3月12日）
- 请你编写代码对沪市指数未来3个月（截止到2020年6月31日）的变化进行预测（将数据转化为按月统计即可）

1990年到2020年沪市指数走势

Summary

- Prophet 针对的是商业预测任务
- 优点：不需要特征工程就能得到趋势，季节因素和节假日因素
- 不 足 ： 无法利用更多的信息，如在预测商品的销量时，无法利用商品的信息，门店的信息，促销的信息等
- 传入prophet的数据分为两列 ds 和 y

ds表示时间戳（pandas的日期格式）

y表示true value，也是需要预测的值（数值型）

- 带holidays参数的prophet

m = Prophet(holidays=holidays)

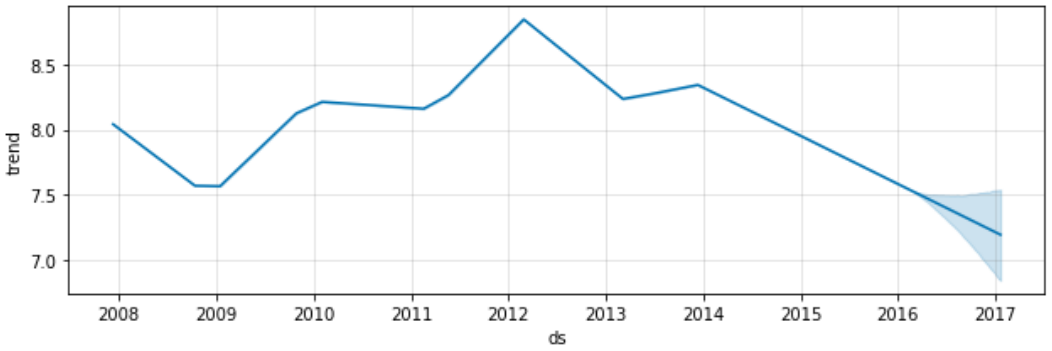
- 生成未来日期

future = model.make_future_dataframe(periods=365)

future为时间轴，在原有基础上增加365天（会包括之前的历史时间戳）

- 模型训练 model.fit(df)
- 模型预测 forecast = model.predict(future)
- 成分分析，plot_components(forecast)

绘制trend, weekly, yearly趋势图，会包括之前历史时间戳预测



Summary

- forecaset字段包括：

ds, 时间轴

trend, trend_lower, trend_upper, 趋势

yhat, yhat_lower, yhat_upper, 预测值

weekly, weekly_lower, weekly_upper, 星期趋势

yearly, yearly_lower, yearly_upper, 年趋势

additive_terms, additive_terms_lower, additive_terms_upper, 加法模型趋势（星期趋势+年趋势），即 forecast['additive_terms'] = forecast['weekly'] + forecast['yearly']

multiplicative_terms, multiplicative_terms_lower, multiplicative_terms_upper，乘法模型趋势，如果使用加法模型时，multiplicative_terms为空

- 趋势变化点 model.changepoints

趋势变化点：时间序列经常会在轨迹中发生突然变化，可以自动检测出这些点

当模型训练完之后，就可以找到趋势变化点，默认为25个，分布在前80%的时间序列中

可以使用参数n_changepoints设置潜在变化点的数量，比如

model= prophet (n_changepoints=30)

可以使用参数changepoint_range设置前多少的时间序列来寻找潜在变化点，比如在时间序列的前90%处寻找潜在的变化点

model = Prophet(changepoint_range=0.9)

人工指定突变点的位置：

model = Prophet(changepoints=['2014-01-01'])

Summary

- 指定预测类型

growth='linear'或growth = "logistic"

默认的增长趋势为linear

如果使用growth="logistic"，就需要指定cap，因为预测时需要用到cap，可以不指定floor，因为logistic默认的最小饱和值是0

m = Prophet(growth='logistic')

df['cap'] = 6 # 不设置会报错

- 模型的学习方式

默认情况下为加性的，如果改成乘性的(multiplicative)，需要设置seasonality_mode='multiplicative'

Prophet中的参数设置：

- Capacity，在增量函数是逻辑回归函数的时候，需要设置的容量值
- Change Points：通过 n_changepoints 和 changepoint_range 来设置时间序列的变化点
- 季节性和节假日，可以根据实际的业务需求来指定相应的节假日
- 光滑参数：

changepoint_prior_scale 设置趋势项的灵活度，即跟随性，默认为0.05，值越大，拟合的跟随性越好，可能会过拟合

seasonality_prior_scale 用来控制季节项的灵活度

holidays_prior_scale 用来控制节假日的灵活度

时间序列AI大赛

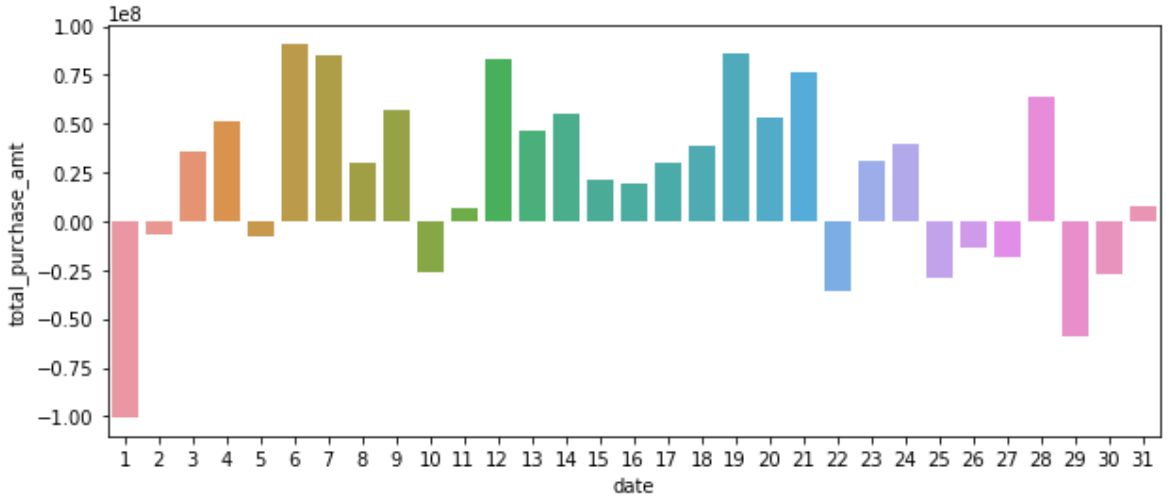
Project：资金流入流出预测

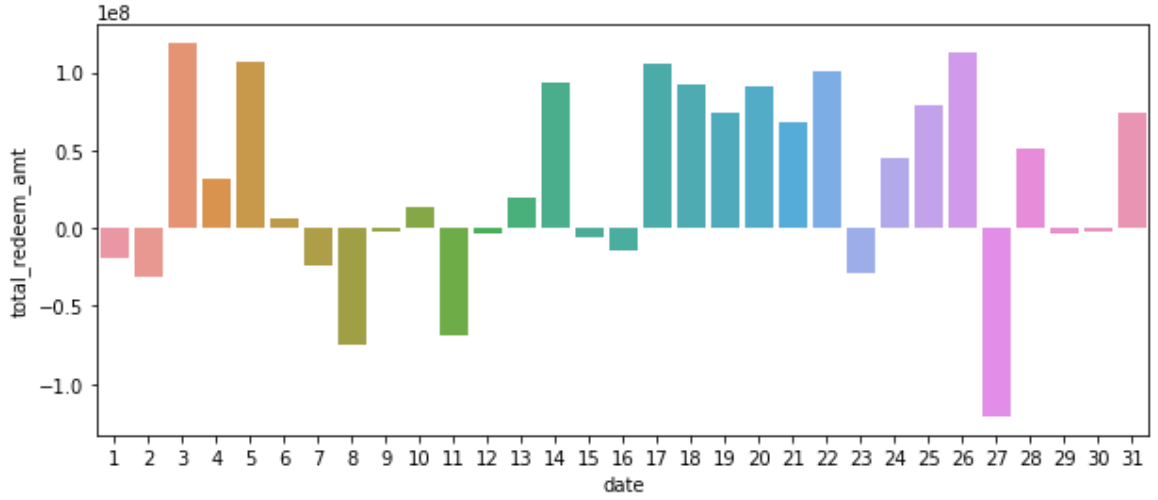
Project：资金流入流出预测

- https://tianchi.aliyun.com/competition/entrance/231573/information
- 数据集一共包括4张表：用户基本信息数据、用户申购赎回数据、收益率表和银行间拆借利率表

2.8万用户，284万行为数据，294天拆解利率，427天收益率

2013-07-01到2014-08-31，预测2014年9月的申购和赎回



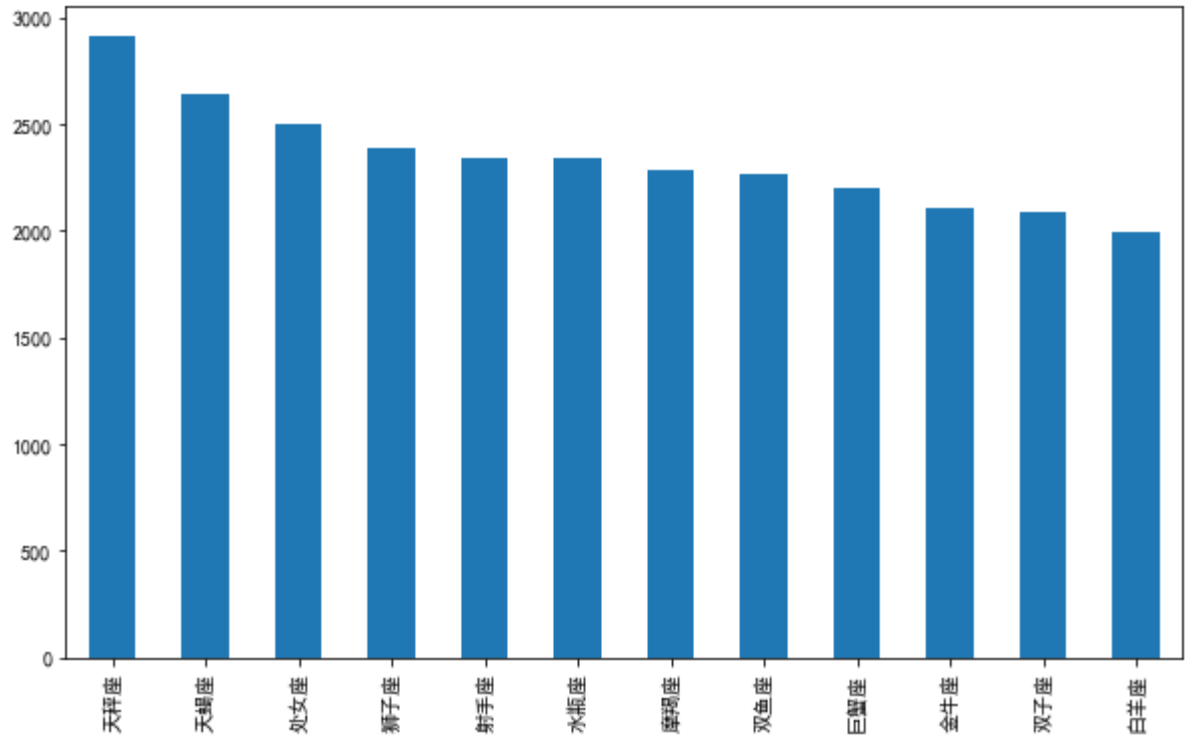


Project：资金流入流出预测

- 用户信息表，user_profile_table

总共随机抽取了约3万用户，主要包含了用户的性别、城市和星座，其中部分用户在 2014 年 9 月份第一次出现，这些用户只在测试数据中

列名	类型	含义	示例
user_id	bigint	用户 ID	1234
Sex	bigint	用户性别（1：男，0：女）	0
City	bigint	所在城市	6081949
constellation	string	星座	射手座



Project：资金流入流出预测

用户申购赎回数据表 user_balance_table

- 数据包括了 20130701 至 20140831 申购和赎回信息，字段包括用户操作时间和操作记录，其中操作记录包括申购和赎回两个部分
- 金额的单位是分，即0.01元
- 如果用户今日消费总量为0，即consume_amt=0，同时四个category字段为空
- 数据经过了脱敏，同时保证了：

今日余额 = 昨日余额 + 今日申购 - 今日赎回，不会出现负值

列名	类型	含义	示例
user_id	bigint	用户id	1234
report_date	string	日期	20140407
tBalance	bigint	今日余额	109004
yBalance	bigint	昨日余额	97389
total_purchase_amt	bigint	今日总购买量=直接购买+收益	21876
direct_purchase_amt	bigint	今日直接购买量	21863
purchase_bal_amt	bigint	今日支付宝余额购买量	0
purchase_bank_amt	bigint	今日银行卡购买量	21863
total_redeem_amt	bigint	今日总赎回量=消费+转出	10261
consume_amt	bigint	今日消费总量	0
transfer_amt	bigint	今日转出总量	10261
tftobal_amt	bigint	今日转出到支付宝余额总量	0
tftocard_amt	bigint	今日转出到银行卡总量	10261
share_amt	bigint	今日收益	13
category1	bigint	今日类目1消费总额	0
category2	bigint	今日类目2消费总额	0
category3	bigint	今日类目3消费总额	0
category4	bigint	今日类目4消费总额	0

Project：资金流入流出预测

收益率表 mfd_day_share_interest

- 收益表为余额宝在 14 个月内的收益率表

列名	类型	含义	示例
mfd_date	string	日期	20140102
mfd_daily_yield	double	万份收益，即1万块钱的收益。	1.5787
mfd_7daily_yield	double	七日年化收益率（%）	6.307

上海银行间同业拆放利率表 mfd_bank_shibor

- 银行间拆借利率表是14个月期间银行之间的拆借利率（皆为年化利率）

列名	类型	含义	示例
mfd_date	String	日期	20140102
Interest_O_N	Double	隔夜利率（%）	2.8
Interest_1_W	Double	1周利率（%）	4.25
Interest_2_W	Double	2周利率（%）	4.9
Interest_1_M	Double	1个月利率（%）	5.04
Interest_3_M	Double	3个月利率（%）	4.91
Interest_6_M	Double	6个月利率（%）	4.79
Interest_9_M	Double	9个月利率（%）	4.76
Interest_1_Y	Double	1年利率（%）	4.78

Project：资金流入流出预测

收益计算方式

- 主要基于实际余额宝收益计算方法，进行了简化

1）收益计算的时间不再是会计日，而是自然日，以0点为分隔（0点之前算昨天，0点之后算今天）

2）收益的显示时间，即实际将第一份收益打入用户账户的时间，以周一转入周三显示为例，如果用户在周一存入10000元，即1000000分，那么这笔金额是周一确认，周二是开始产生收益，在周三将周二产生的收益打入到用户的账户中，此时用户的账户中显示的是1000110分

转入时间	首次显示收益时间
周一	周三
周二	周四
周三	周五
周四	周六
周五	下周二
周六	下周三
周天	下周三

Project：资金流入流出预测

提交结果表 tc_comp_predict_table

字段	类型	含义	示例
report_date	bigint	日期	20140901
purchase	bigint	申购总额	40000000
redeem	bigint	赎回总额	30000000

每一行数据是一天对申购、赎回总额的预测值，输出2014年9月每天的预测，共30行。 purchase 和 redeem 都是金额数据，精确到分

输出示意：

20140901	40000000	30000000
20140902	40000000	30000000
20140903	40000000	30000000

评估指标：

1）计算测试集上每天的申购及赎回与实际的误差

每日申购相对误差(真实值z_i，预测值为$\hat{z}_i$)：

$$\text{Purchase}_i = \frac{|z_i - \hat{z}_i|}{z_i}$$

每日赎回相对误差(真实值y_i，预测值为$\hat{y}_i$)：

$$\text{Redeem}_i = \frac{|y_i - \hat{y}_i|}{y_i}$$

2）误差与得分之间的计算公式不公布，但保证单调递减

第 i 天的申购误差Purchasei=0 ，这一天的得分为10分；

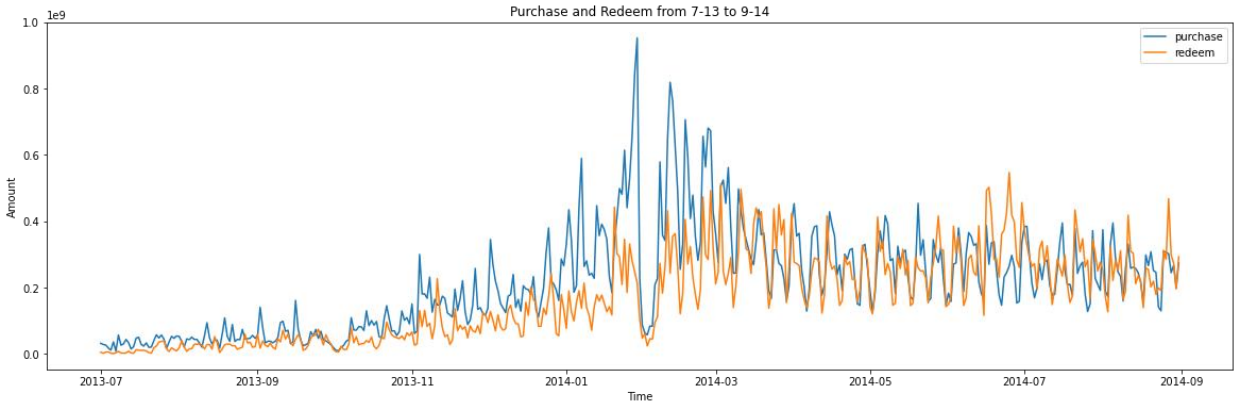
当Purchasei ＞ 0.3，得分为0

3）总积分 = 申购预测得分*45% + 赎回预测得分*55%

Project：资金流入流出预测

- 数据探索EDA

1）每日总购买与赎回量的时间序列图



2）STL分解，将时序图拆分为：Trend + Seasonal + Residual

- 方法1：采用时间序列进行预测

Step1，平稳性检测 adfuller

Step2，采用 ARIMA模型

Step3，模型训练集与预测

- 方法2：基于时序规则的挖掘

Step1，获得周期因子（weekday）

Step2，计算base

Step3，使用base * 周期因子进行预测

Project：资金流入流出预测

- read_csv中的日期格式解析

pd.read_csv('user_balance_table.csv', parse_dates = ['report_date'])

设置parse_dates参数，将时间字符串转换为日期格式

- DataFrame.diff()函数

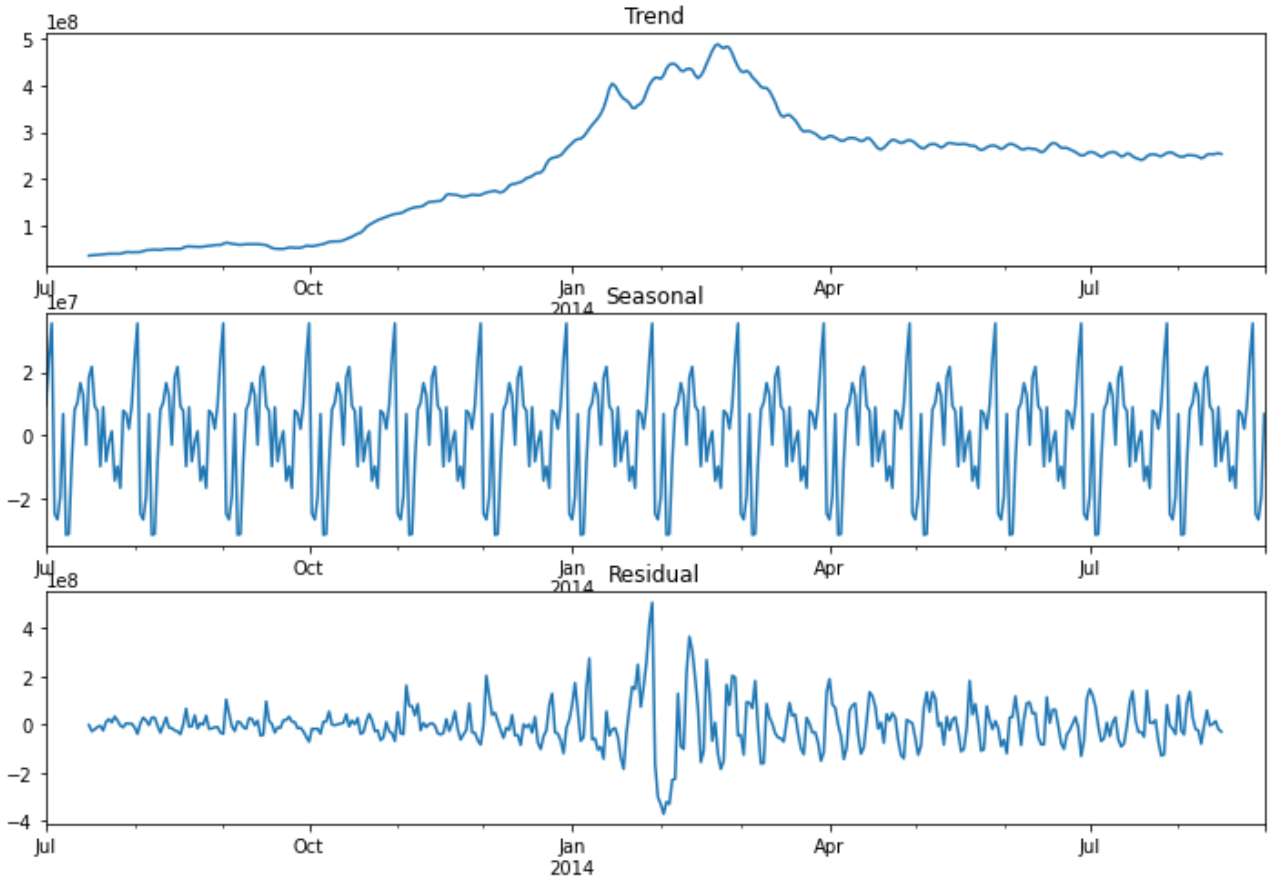
用来将数据进行某种移动之后与原数据进行比较得出的差异数据

- DataFrame.shift()函数

可以把数据移动指定的位数

periods=-1 往上移动或往左移动

periods=1 往下移动或往右移动



Project：资金流入流出预测

平稳性检测（ADF检测）：

- 在使用时间序列模型时（比如 ARMA、ARIMA），需要时间序列是平稳的，所以第一步都需要进行平稳性检验，常用的统计检验方法为ADF检验（也称为单位根检验）
- ADF检验，就是判断序列是否存在单位根，如果序列平稳，就不存在单位根，否则，就会存在单位根
- ADF检验的 H0 假设就是存在单位根，如果得到的显著性检验统计量小于三个置信度（10%，5%，1%），则对应有（90%，95，99%）的把握来拒绝原假设

```
from statsmodels.tsa.stattools import adfuller

t=adfuller(df_p['total_purchase_amt'])
```

```
(-1.5898802926313507, 0.4886749751375928, 18, 408, {'1%': -3.446479704252724, '5%': -2.8686500930967354, '10%': -2.5705574627547096}, 15960.28197033403)
```

输出结果依次为：

t-statistic, p-value, usedlag, nobs

critical-value：测试统计数据的临界值为1%，5%和10%

AIC

如何确定该序列能否平稳：

主要看1%、%5、%10不同程度拒绝原假设的统计值和ADF Test result的比较，如果ADF Test result同时小于1%、5%、10%即说明非常好地拒绝原假设（原假设是不稳定的，因此证明是平稳的）

这里，adf结果为-1.58988， 大于三个level的统计值，无法拒绝原假设（原假设是不平稳的），需要进行一阶差分后，再进行检验

Project：资金流入流出预测

- 时间序列预测

1）针对购买purchase建模

ARIMA(purchase,order=(7,1,5)).fit()

model.predict('2014-09-01', '2014-09-30',typ='levels')

2）针对赎回redeem建模

ARIMA(redeem,order=(7,1,5)).fit()

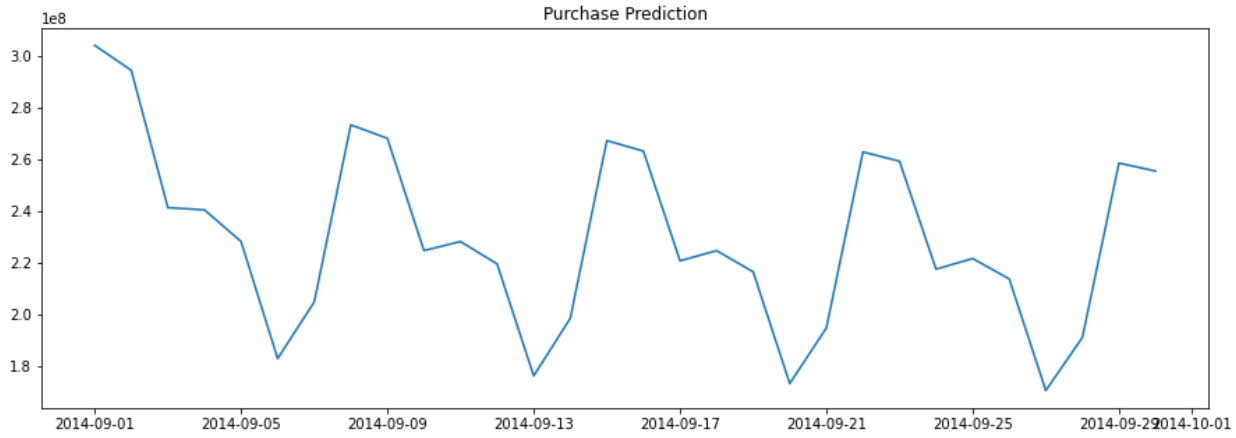
model.predict('2014-09-01', '2014-09-30',typ='levels')

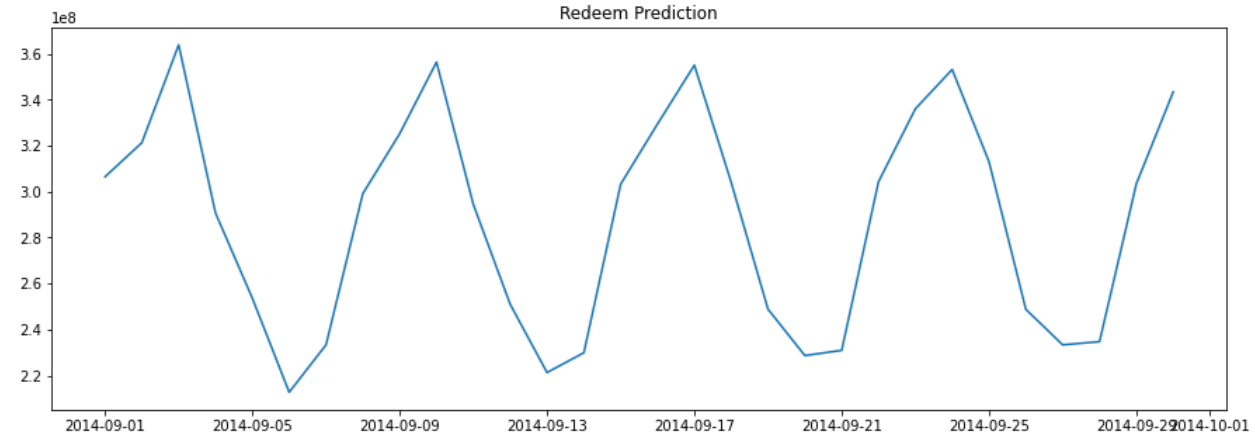
Thinking：模型预测准确性如何？

1）过于简单，实际情况并不是

2）周一到周日的特征规律没有利用

3）没有考虑特殊时间，比如节日，利率波动节点





Project：资金流入流出预测

时间序列规则：

- 选择特征

可以用简单的统计量来作为特征，从中提取出有用的信息

1）中位数：居于中间位置的数，较为稳健

2）均值：当分布符合正态分布时，可以代表整体特征

3）临近数据：离待测数据越近的数据对其影响越大

基于周期因子的时间序列预测

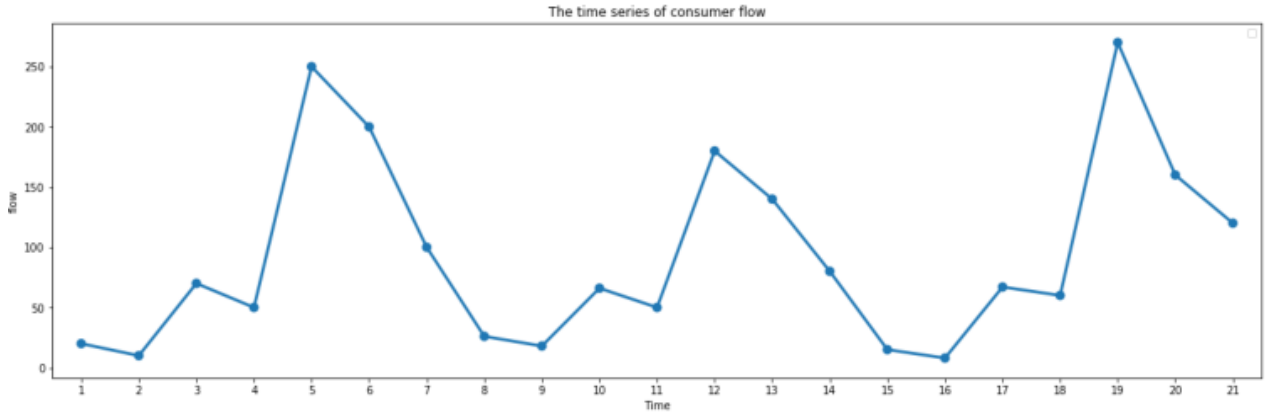
- 很多数据都具有明显的周期性，比如客流量，支付等
- 需要确定周期长度，比如一周7天，一个月30天，结合STL分解 (Seasonal and Trend decomposition) 观察周期变化，缺点是没有考虑到节假日、突发事件等情况

Project：资金流入流出预测

基于周期因子的时间序列预测

- 假设给任务是根据前三周的数据预测第四周每天的客流量

	周一	周二	周三	周四	周五	周六	周日	周均值
第一周	20	10	70	50	250	200	100	100
第二周	26	18	66	50	180	140	80	80
第三周	15	8	67	60	270	160	120	100



Step 1，获得周期因子（weekday）

获得星期几的均值，再除以整体均值

	周一	周二	周三	周四	周五	周六	周日
第一周	20	10	70	50	250	200	100
第二周	26	18	66	50	180	140	80
第三周	15	8	67	60	270	160	120
均值	20.33	12	67.67	53.33	233.33	166.67	100
因子	0.22	0.13	0.73	0.57	2.50	1.79	1.07

Step 2，计算base

Step3，使用base * 周期因子进行预测

假设base=100，可以得到第四周的客流量

	周一	周二	周三	周四	周五	周六	周日
第四周	22	13	73	57	250	179	107

Project：资金流入流出预测

预测下个月每一天的情况：

- 如果想预测下个月每天的流量情况，可以基于每月的规律（1-30号的平均流量）* 周期因子

Step1，计算周期因子（weekday）

Step2，计算每日（1号-30号）均值，即1号的平均流量，2号的平均流量 …

Step3，统计星期几（weekday）在每日（day）出现的频次

Step4，基于周期因子获得加权均值，得到每日的base（去掉周期因子的影响）

Step5，根据每日的base和周期因子进行预测

Step 1，获得周期因子（weekday）

Project：资金流入流出预测

- 观察特殊日期

清明节，2014年4月5-7日

五一，2014年5月1-3日

六一，2014年5月31-6月2日

中秋节：2014年9月6-8日

国庆节：2014年10月1-7日

打卡：资金流入流出预测

针对AI大赛：资金流入流出预测，编写AI算法，进行预测，挑战分数 > 120

https://tianchi.aliyun.com/competition/entrance/231573

数据集 Purchase Redemption Data.zip

- 选择适合的时间范围
- 时间序列模型

ARMA/ARIMA

prophet

- 周期因子分析
- 模型融合


Thank You
Using data to solve problems