数字文创会展运营数据分析

面向 IP 展示、文创市集、互动体验等业态，用 Python 采集公开数据并进行分析与可视化。

数据来源

数据	来源
中国服务业增加值（年度）	世界银行 World Bank API
多品类中文评论 online_shopping_10_cats	公开中文 NLP 语料
Online Retail 交易明细	UCI 机器学习库
Bike Sharing 逐时计数	UCI 机器学习库

任务模块（6 个）：市场规模分析、任务与资源分配、社交媒体情感分析、参展商与观众管理、场地与设施管理、会后评估。

In [1]:
```
# 导入库与中文字体设置
import warnings, logging, os, json, ssl, zipfile, urllib.request
warnings.filterwarnings('ignore')
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_theme(style='whitegrid')
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei', 'SimHei']
plt.rcParams['axes.unicode_minus'] = False
ssl._create_default_https_context = ssl._create_unverified_context

DATA_DIR = 'data'
os.makedirs(DATA_DIR, exist_ok=True)
UA = {'User-Agent': 'Mozilla/5.0'}

def fetch(url, fname, timeout=180):
    path = os.path.join(DATA_DIR, fname)
    if not os.path.exists(path) or os.path.getsize(path) == 0:
        print('下载', fname)
        req = urllib.request.Request(url, headers=UA)
        with urllib.request.urlopen(req, timeout=timeout) as r, open(path, 'wb') as
            f.write(r.read())
    return path

print('环境就绪')
```

环境就绪

一、会展行业市场规模分析

文创会展依托服务业景气。采集世界银行「中国服务业增加值」年度数据，用 Holt 指数平滑预测未来三年。

In [2]:
```
# 采集世界银行数据：中国服务业增加值
def fetch_worldbank(indicator, fname):
    path = os.path.join(DATA_DIR, fname)
    if not os.path.exists(path) or os.path.getsize(path) == 0:
        url = 'https://api.worldbank.org/v2/country/CHN/indicator/' + indicator + '
        req = urllib.request.Request(url, headers=UA)
        with urllib.request.urlopen(req, timeout=60) as r:
            js = json.loads(r.read().decode('utf-8'))
        recs = [(int(d['date']), d['value']) for d in js[1] if d['value'] is not No
        pd.DataFrame(sorted(recs), columns=['年份', '指标值']).to_csv(path, index=Fa
    return pd.read_csv(path)

serv = fetch_worldbank('NV.SRV.TOTL.CD', 'worldbank_services.csv')
serv = serv[serv['年份'] >= 2004].reset_index(drop=True)
serv['服务业增加值_万亿美元'] = (serv['指标值'] / 1e12).round(3)
print('年份范围', serv['年份'].min(), '-', serv['年份'].max(), '共', len(serv), '条')
serv[['年份', '服务业增加值_万亿美元']].tail()
```

```
年份范围 2004 - 2024 共 21 条
```

Out[2]:

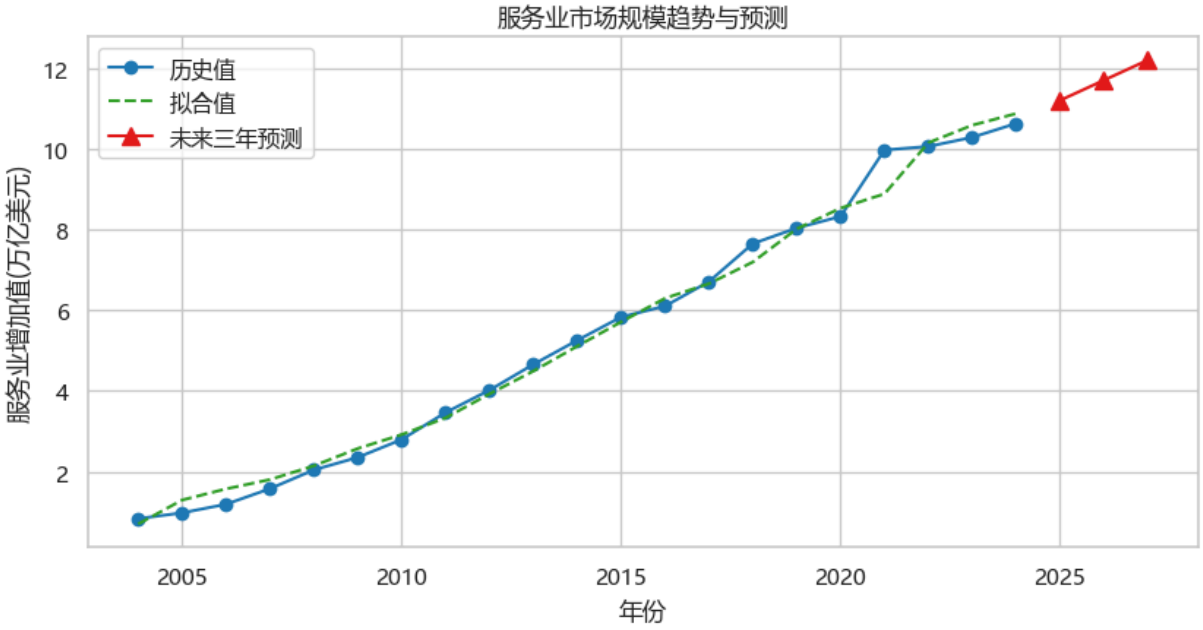
	年份	服务业增加值_万亿美元
16	2020	8.330
17	2021	9.978
18	2022	10.066
19	2023	10.293
20	2024	10.637

In [3]:
```
# Holt 指数平滑预测
from statsmodels.tsa.holtwinters import ExponentialSmoothing
val = serv['服务业增加值_万亿美元'].values
fit = ExponentialSmoothing(pd.Series(val), trend='add', initialization_method='esti
fc = fit.forecast(3)
last_year = int(serv['年份'].max())
future_years = np.arange(last_year + 1, last_year + 4)
cagr = (val[-1] / val[0]) ** (1 / (len(val) - 1)) - 1
print('年均增长率 CAGR =', round(cagr * 100, 2), '%')
print(pd.DataFrame({'年份': future_years, '预测值_万亿美元': np.round(fc.values, 3)})
```

```
年均增长率 CAGR = 13.57 %
   年份  预测值_万亿美元
2025    11.209
2026    11.710
2027    12.210
```

In [4]:
```
# 可视化趋势与预测
yrs = serv['年份'].values
plt.figure(figsize=(8.5, 4.6))
plt.plot(yrs, val, '-o', color='#1f78b4', label='历史值')
plt.plot(yrs, fit.fittedvalues, '--', color='#33a02c', label='拟合值')
```

```
plt.plot(future_years, fc.values, '-^', color='#e31a1c', ms=9, label='未来三年预测')
plt.xlabel('年份')
plt.ylabel('服务业增加值(万亿美元)')
plt.title('服务业市场规模趋势与预测')
plt.legend()
plt.tight_layout()
plt.show()
```



服务业增加值稳步上升，指数平滑预测未来三年继续增长，说明文创会展市场空间较大。

二、会展运营任务与资源分配

把筹办任务分给若干运营组，让各组负荷尽量均衡。先用 LPT 贪心分配，再做局部搜索改进。

In [5]:
```
# 任务工作量
task_names = ['IP招商', '舞美搭建', '票务系统', '新媒体运营', '志愿者培训', '安保方案',
              '餐饮协调', '非遗招募', '互动装置', '直播策划', '物料印刷', '嘉宾邀约']
task_cost = [9, 11, 6, 8, 5, 7, 6, 8, 10, 7, 4, 9]
M = 4
groups = ['运营组甲', '运营组乙', '运营组丙', '运营组丁']
print('任务总工作量', sum(task_cost), '人天，理论均衡负荷', sum(task_cost) / M)
pd.DataFrame({'任务': task_names, '工作量_人天': task_cost})
```

```
任务总工作量 90 人天，理论均衡负荷 22.5
```

Out[5]:

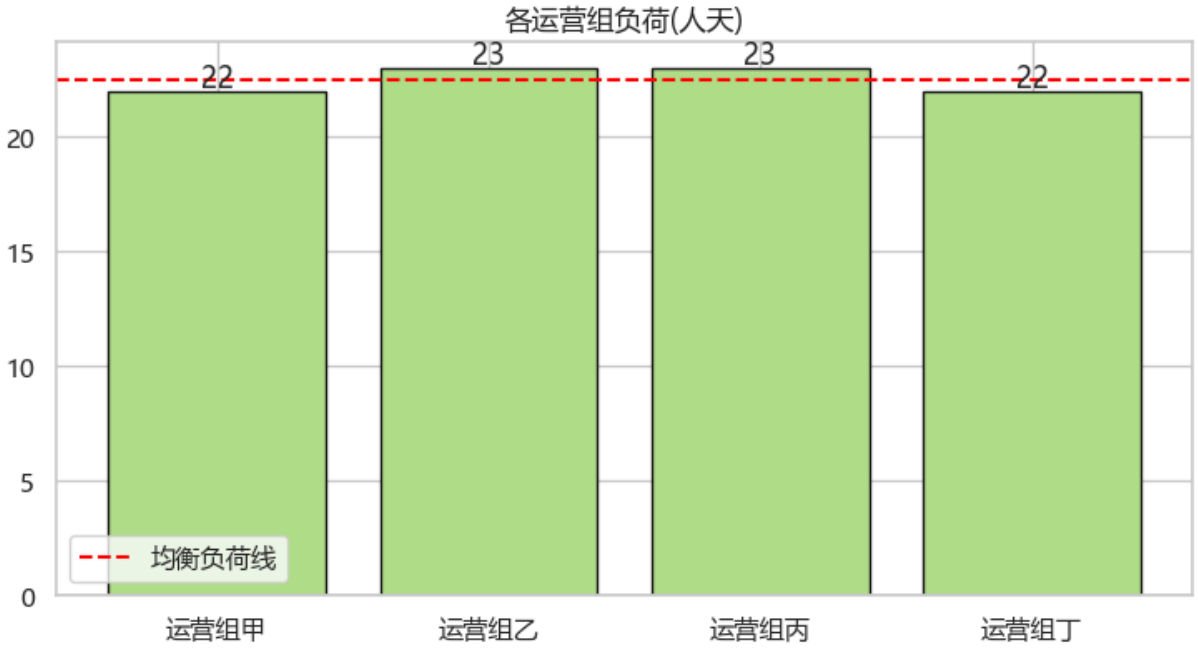
	任务	工作量_人天
0	IP招商	9
1	舞美搭建	11
2	票务系统	6
3	新媒体运营	8
4	志愿者培训	5
5	安保方案	7
6	餐饮协调	6
7	非遗招募	8
8	互动装置	10
9	直播策划	7
10	物料印刷	4
11	嘉宾邀约	9

In [6]:
```
# LPT 贪心加局部搜索
order = sorted(range(len(task_names)), key=lambda i: -task_cost[i])
assign = [0] * len(task_names)
loads = [0] * M
for i in order:
    g = int(np.argmin(loads))
    assign[i] = g
    loads[g] += task_cost[i]
print('贪心后最大负荷', max(loads), '人天')

improved = True
while improved:
    improved = False
    for i in range(len(task_names)):
        for g in range(M):
            if g == assign[i]:
                continue
            new_loads = loads.copy()
            new_loads[assign[i]] -= task_cost[i]
            new_loads[g] += task_cost[i]
            if max(new_loads) < max(loads):
                loads = new_loads
                assign[i] = g
                improved = True
print('局部搜索后最大负荷', max(loads), '人天')
final_loads = loads
```

```
贪心后最大负荷 23 人天
局部搜索后最大负荷 23 人天
```

In [7]:
```
# 各组负荷与分配结果
plt.figure(figsize=(7.5, 4.2))
```

```
bars = plt.bar(groups, final_loads, color='#b2df8a', edgecolor='k')
plt.axhline(sum(task_cost) / M, color='red', ls='--', label='均衡负荷线')
for b, v in zip(bars, final_loads):
    plt.text(b.get_x() + b.get_width() / 2, v + 0.2, str(v), ha='center')
plt.title('各运营组负荷(人天)')
plt.legend()
plt.tight_layout()
plt.show()

pd.DataFrame({'任务': task_names, '工作量_人天': task_cost,
              '分配小组': [groups[g] for g in assign]}).sort_values('分配小组')
```



Out[7]:

	任务	工作量_人天	分配小组
4	志愿者培训	5	运营组丁
7	非遗招募	8	运营组丁
11	嘉宾邀约	9	运营组丁
0	IP招商	9	运营组丙
3	新媒体运营	8	运营组丙
6	餐饮协调	6	运营组丙
2	票务系统	6	运营组乙
5	安保方案	7	运营组乙
8	互动装置	10	运营组乙
1	舞美搭建	11	运营组甲
9	直播策划	7	运营组甲
10	物料印刷	4	运营组甲

局部搜索进一步压低了最大负荷，四个组的工作量比较均衡。

三、会展社交媒体情感分析

采集 online_shopping_10_cats 多品类中文评论，用情感词典法（情感词加程度词、否定词）给评论打分，并用语料自带标签检验效果。

In [8]:
```
# 采集评论并准备词典
import jieba
jieba.setLogLevel(logging.INFO)
shop_url = 'https://hf-mirror.com/datasets/dirtycomputer/online_shopping_10_cats/re
shop_path = fetch(shop_url, 'online_shopping_10_cats.csv')
df_full = pd.read_csv(shop_path).dropna(subset=['review'])
df_shop = (df_full.groupby(['cat', 'label'], group_keys=False)
           .apply(lambda g: g.sample(min(len(g), 350), random_state=42)).reset_inde
print('评论条数', len(df_shop), '品类数', df_shop['cat'].nunique(), '正面占比', round

pos_dict = {'好': 1, '不错': 1.5, '满意': 1.5, '喜欢': 1.5, '推荐': 1.5, '赞': 1.5, '
            '优秀': 2, '完美': 2, '惊喜': 1.5, '舒服': 1.2, '快': 1, '正品': 1.2, '清
neg_dict = {'差': -1.5, '失望': -2, '垃圾': -2, '坑': -2, '慢': -1, '贵': -1, '烂': -
            '问题': -1, '后悔': -2, '糟糕': -2, '卡': -1, '假': -1.8, '难用': -1.5, '
degree = {'很': 1.5, '非常': 2.0, '特别': 1.8, '超': 1.8, '太': 1.6, '十分': 1.8, '有
negation = set(['不', '没', '没有', '无', '别'])
```

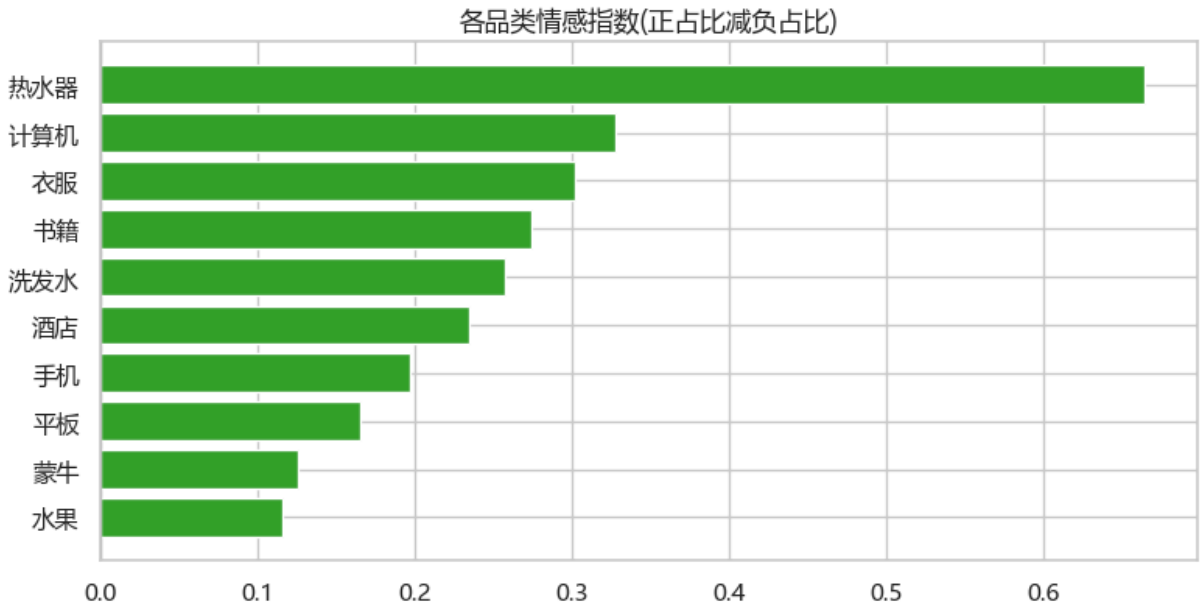
```
评论条数 6750 品类数 10 正面占比 51.9 %
```

In [9]:
```
# 词典打分与极性判定
def sentiment_score(text):
    words = jieba.lcut(str(text))
    score = 0.0
    for i, w in enumerate(words):
        val = pos_dict.get(w, neg_dict.get(w, 0))
        if val != 0:
            weight = 1.0
            neg = 1
            for j in range(max(0, i - 2), i):
                if words[j] in degree:
                    weight = weight * degree[words[j]]
                if words[j] in negation:
                    neg = neg * -1
            score = score + val * weight * neg
    return score

df_shop['情感得分'] = df_shop['review'].map(sentiment_score)
df_shop['预测极性'] = np.where(df_shop['情感得分'] > 0, 1, np.where(df_shop['情感得分
covered = df_shop[df_shop['预测极性'] != -1]
acc = (covered['预测极性'] == covered['label']).mean()
print('可判定覆盖率', round(len(covered) / len(df_shop) * 100, 1), '%')
print('可判定样本准确率', round(acc * 100, 1), '%')
```

```
可判定覆盖率 59.1 %
可判定样本准确率 86.3 %
```

In [10]:

```
# 各品类情感指数与词云
from wordcloud import WordCloud
cat_index = df_shop.groupby('cat').apply(lambda g: (g['情感得分'] > 0).mean() - (g['
plt.figure(figsize=(8, 4.2))
plt.barh(cat_index.index, cat_index.values, color=np.where(cat_index.values >= 0, '
plt.axvline(0, color='gray')
plt.title('各品类情感指数(正占比减负占比)')
plt.tight_layout()
plt.show()

pos_tokens = ' '.join(w for t in df_shop[df_shop['label'] == 1]['review'].head(1500
                      for w in jieba.lcut(str(t)) if w in pos_dict)
try:
    wc = WordCloud(font_path=r'C:\Windows\Fonts\msyh.ttc', width=700, height=320,
                   background_color='white', collocations=False).generate(pos_token
    plt.figure(figsize=(8, 3.6))
    plt.imshow(wc)
    plt.axis('off')
    plt.title('正面评论高频词')
    plt.tight_layout()
    plt.show()
except Exception as e:
    print('词云跳过', e)
```



情感词典法在可判定的评论上与真实标签基本一致；不同品类的情感指数有差异，可用于分业态的口碑参考。

四、会展参展商与观众管理

把交易客户看作参展观众，采集 Online Retail 数据构建 RFM 指标做 KMeans 分群，再用决策树根据首单特征预测是否复购。

In [11]:
```python
# 采集交易数据并构建 RFM
def fetch_online_retail():
    csvp = os.path.join(DATA_DIR, 'online_retail_clean.csv')
    if os.path.exists(csvp):
        return pd.read_csv(csvp, parse_dates=['InvoiceDate'])
    zp = fetch('https://archive.ics.uci.edu/static/public/352/online+retail.zip', '
    with zipfile.ZipFile(zp) as z:
        name = [n for n in z.namelist() if n.lower().endswith(('.xlsx', '.xls'))][0
        with z.open(name) as f:
            df = pd.read_excel(f)
    df = df.dropna(subset=['CustomerID'])
    df = df[(df['Quantity'] > 0) & (df['UnitPrice'] > 0)]
    df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
    df['金额'] = df['Quantity'] * df['UnitPrice']
    df.to_csv(csvp, index=False)
    return df

retail = fetch_online_retail()
snapshot = retail['InvoiceDate'].max() + pd.Timedelta(days=1)
rfm = retail.groupby('CustomerID').agg(
    R=('InvoiceDate', lambda s: (snapshot - s.max()).days),
    F=('InvoiceNo', 'nunique'),
    M=('金额', 'sum')).reset_index()
rfm = rfm[(rfm['M'] > 0) & (rfm['M'] < rfm['M'].quantile(0.99))]
print('客户数', len(rfm))
rfm[['R', 'F', 'M']].describe().round(1)
```

客户数 4294

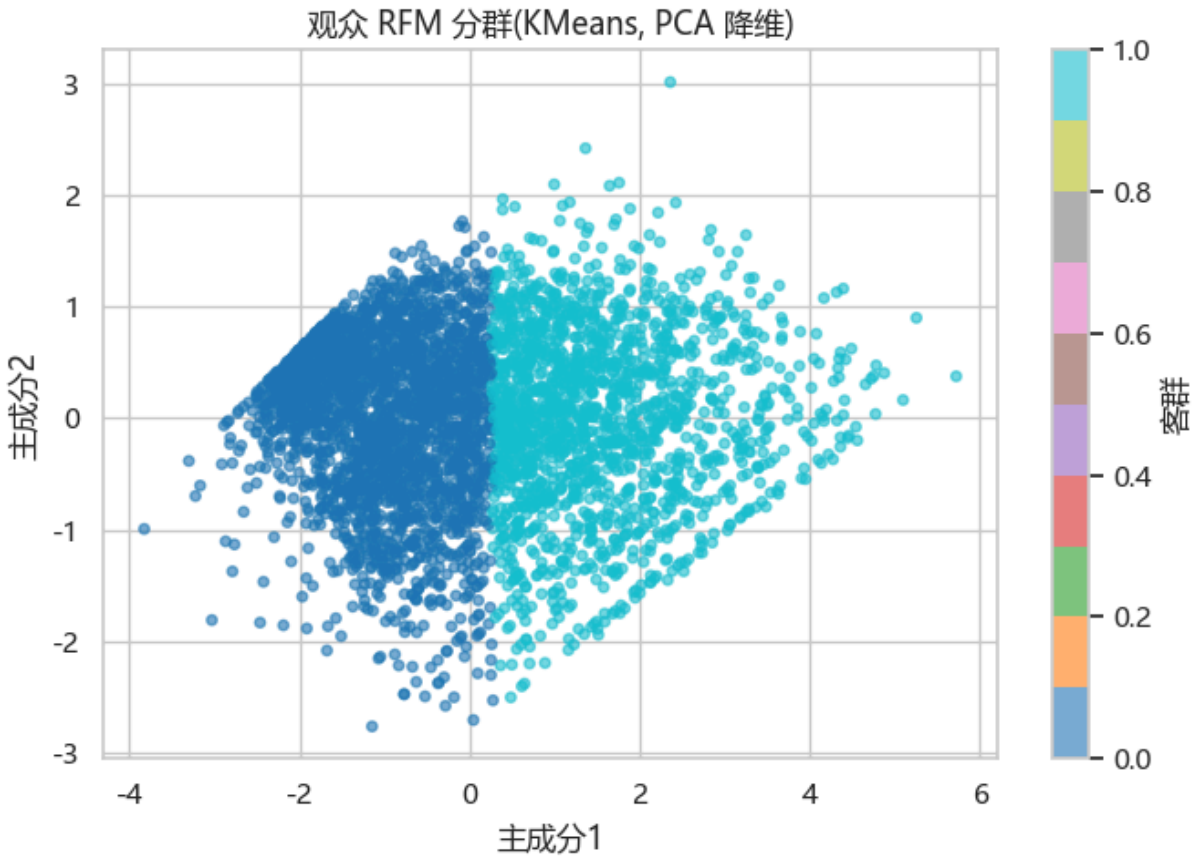
Out[11]:

	R	F	M
count	4294.0	4294.0	4294.0
mean	93.2	3.9	1411.2
std	100.1	4.9	2108.8
min	1.0	1.0	3.8
25%	18.0	1.0	305.9
50%	51.0	2.0	663.7
75%	143.0	4.8	1612.4
max	374.0	93.0	19824.0

In [12]:

```
# KMeans 客户分群
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.decomposition import PCA

Xrfm = np.log1p(rfm[['R', 'F', 'M']].values)
Xs = StandardScaler().fit_transform(Xrfm)
sils = {}
for k in range(2, 6):
    km = KMeans(n_clusters=k, random_state=42, n_init=10).fit(Xs)
    sils[k] = round(silhouette_score(Xs, km.labels_), 3)
best_k = max(sils, key=sils.get)
print('各 k 轮廓系数', sils, '最优 k =', best_k)

km = KMeans(n_clusters=best_k, random_state=42, n_init=10).fit(Xs)
rfm['客群'] = km.labels_
pca = PCA(2).fit_transform(Xs)
plt.figure(figsize=(7, 5))
sc = plt.scatter(pca[:, 0], pca[:, 1], c=km.labels_, cmap='tab10', alpha=0.6, s=15)
plt.title('观众 RFM 分群(KMeans, PCA 降维)')
plt.xlabel('主成分1')
plt.ylabel('主成分2')
plt.colorbar(sc, label='客群')
plt.tight_layout()
plt.show()
rfm.groupby('客群')[['R', 'F', 'M']].mean().round(1)
```

各 k 轮廓系数 {2: 0.433, 3: 0.329, 4: 0.335, 5: 0.314} 最优 k = 2



Out[12]:

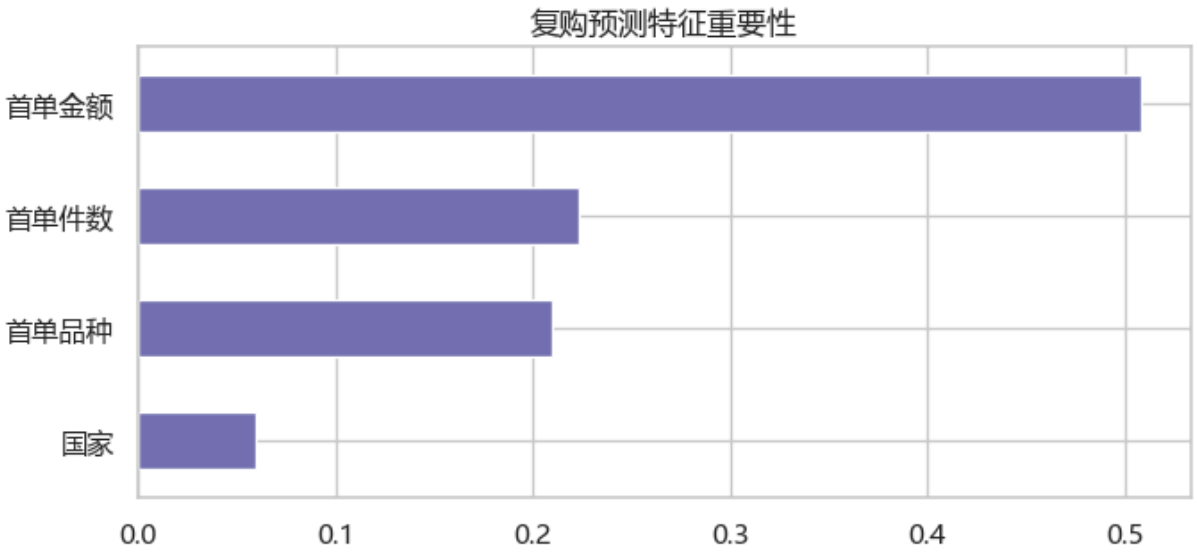
客群	R	F	M
0	136.0	1.6	473.2
1	27.8	7.3	2845.4

In [13]:

```
# 决策树预测复购
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder

first = retail.sort_values('InvoiceDate').groupby('CustomerID').first()
first_inv = retail.sort_values('InvoiceDate').groupby(['CustomerID', 'InvoiceNo']).
first_order = first_inv.groupby('CustomerID').first()
agg_first = retail.merge(first_order[['InvoiceNo']], on=['CustomerID', 'InvoiceNo']
fo = agg_first.groupby('CustomerID').agg(首单金额=('金额', 'sum'), 首单件数=('Quantit
                                         首单品种=('StockCode', 'nunique')).reset_i
fo['国家'] = LabelEncoder().fit_transform(first.loc[fo['CustomerID'], 'Country'].va
ninv = retail.groupby('CustomerID')['InvoiceNo'].nunique()
fo['复购'] = (fo['CustomerID'].map(ninv) >= 2).astype(int)

feats = ['首单金额', '首单件数', '首单品种', '国家']
Xtr, Xte, ytr, yte = train_test_split(fo[feats], fo['复购'], test_size=0.3, random_
dt = DecisionTreeClassifier(max_depth=5, random_state=42).fit(Xtr, ytr)
print('复购预测准确率', round(accuracy_score(yte, dt.predict(Xte)), 3), '复购率', rou
imp = pd.Series(dt.feature_importances_, index=feats).sort_values()
```

```
plt.figure(figsize=(7, 3.4))
imp.plot(kind='barh', color='#7570b3')
plt.title('复购预测特征重要性')
plt.tight_layout()
plt.show()
```

复购预测准确率 0.65 复购率 65.6 %



客户可分成高价值、潜力和流失等客群；决策树显示首单金额和品种数对复购影响较大。

五、会展场地与设施管理

采集 Bike Sharing 逐时计数作为客流代理，用随机森林按时段、星期、天气等预测客流，并画分时段热力图。

In [14]:
```
# 采集逐时客流数据
def fetch_bike():
    csvp = os.path.join(DATA_DIR, 'bike_hour.csv')
    if os.path.exists(csvp):
        return pd.read_csv(csvp)
    zp = fetch('https://archive.ics.uci.edu/static/public/275/bike+sharing+dataset.
    with zipfile.ZipFile(zp) as z:
        with z.open('hour.csv') as f:
            df = pd.read_csv(f)
    df.to_csv(csvp, index=False)
    return df

bike = fetch_bike()
bike = bike.rename(columns={'cnt': '客流'})
print('逐时记录数', len(bike))
bike[['hr', 'weekday', 'workingday', 'weathersit', 'temp', '客流']].head()
```

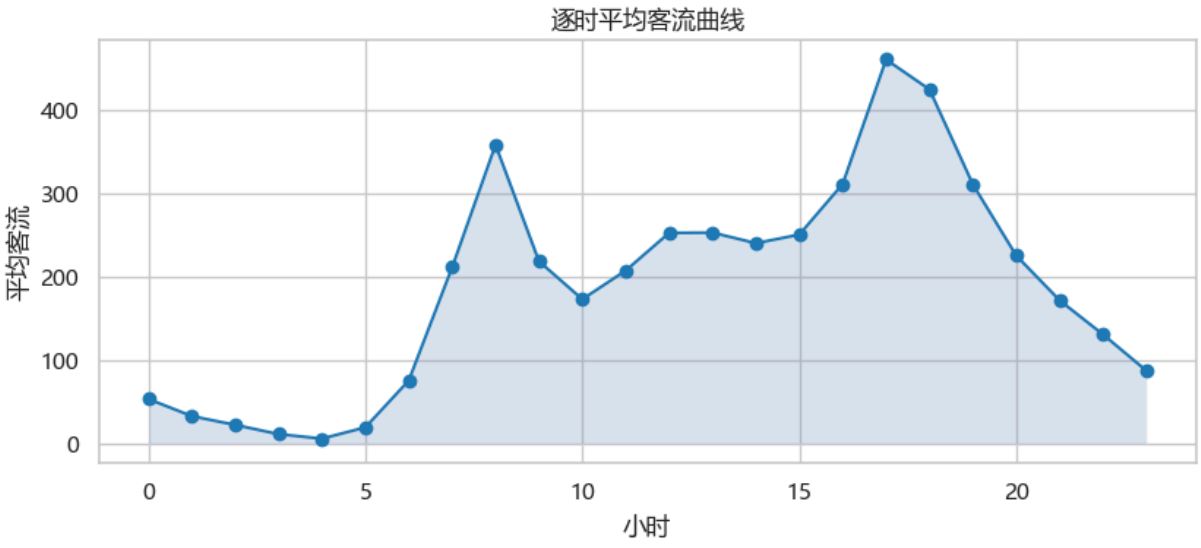
逐时记录数 17379

Out[14]:

	hr	weekday	workingday	weathersit	temp	客流
0	0	6	0	1	0.24	16
1	1	6	0	1	0.22	40
2	2	6	0	1	0.22	32
3	3	6	0	1	0.24	13
4	4	6	0	1	0.24	1

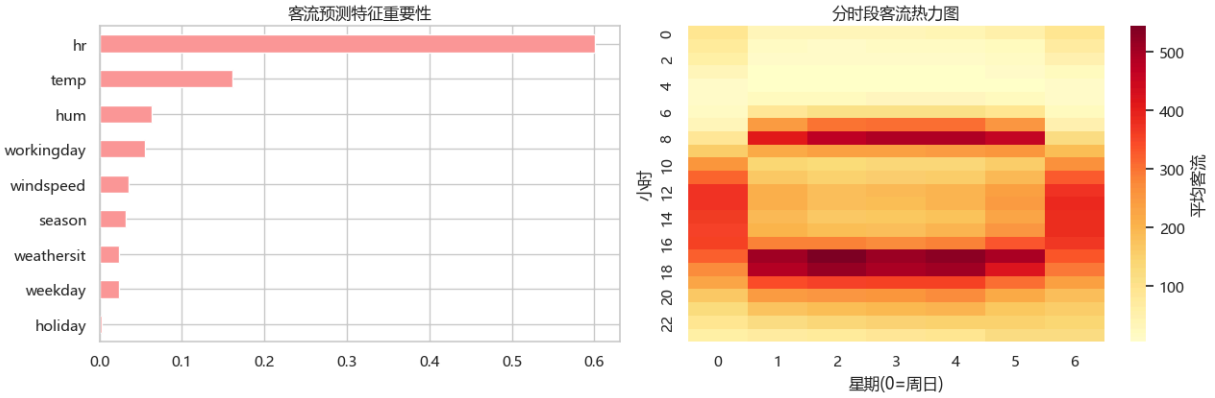
In [15]:
```
# 随机森林客流预测
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_absolute_error

feat = ['hr', 'weekday', 'workingday', 'holiday', 'weathersit', 'temp', 'hum', 'win
X = bike[feat]
y = bike['客流']
Xtr, Xte, ytr, yte = train_test_split(X, y, test_size=0.25, random_state=42)
rf = RandomForestRegressor(n_estimators=200, random_state=42, n_jobs=-1).fit(Xtr, y
pred = rf.predict(Xte)
print('客流预测 R2 =', round(r2_score(yte, pred), 3), 'MAE =', round(mean_absolute_e

hourly = bike.groupby('hr')['客流'].mean()
plt.figure(figsize=(8.5, 4))
plt.plot(hourly.index, hourly.values, '-o', color='#1f78b4')
plt.fill_between(hourly.index, hourly.values, alpha=0.2)
plt.xlabel('小时')
plt.ylabel('平均客流')
plt.title('逐时平均客流曲线')
plt.tight_layout()
plt.show()
```

```
客流预测 R2 = 0.855 MAE = 44.4
```



In [16]:
```
# 特征重要性与客流热力图
imp = pd.Series(rf.feature_importances_, index=feat).sort_values()
fig, ax = plt.subplots(1, 2, figsize=(13, 4.4))
```

```
imp.plot(kind='barh', color='#fb9a99', ax=ax[0])
ax[0].set_title('客流预测特征重要性')
pivot = bike.pivot_table('客流', 'hr', 'weekday', aggfunc='mean')
sns.heatmap(pivot, cmap='YlOrRd', ax=ax[1], cbar_kws={'label': '平均客流'})
ax[1].set_xlabel('星期(0=周日)')
ax[1].set_ylabel('小时')
ax[1].set_title('分时段客流热力图')
plt.tight_layout()
plt.show()
```



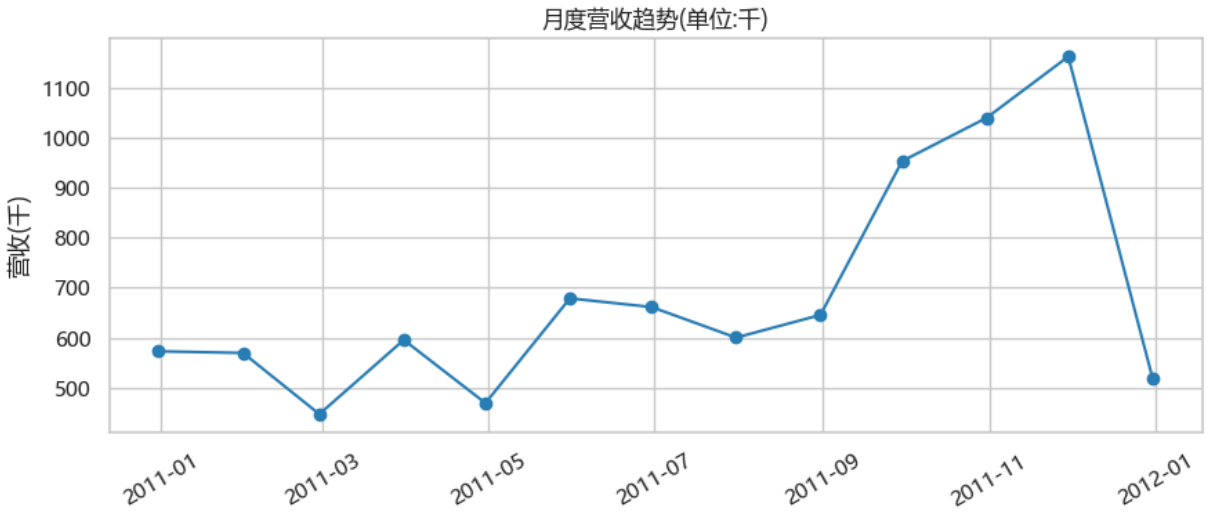
随机森林预测客流效果较好，时段是最重要的因素；热力图显示高峰时段明显，可据此安排现场分流。

六、会展后评估与数据分析

用交易数据算复购率、客单价、覆盖国家、营收增速等经营 KPI，并用雷达图对比目标与实际达成。

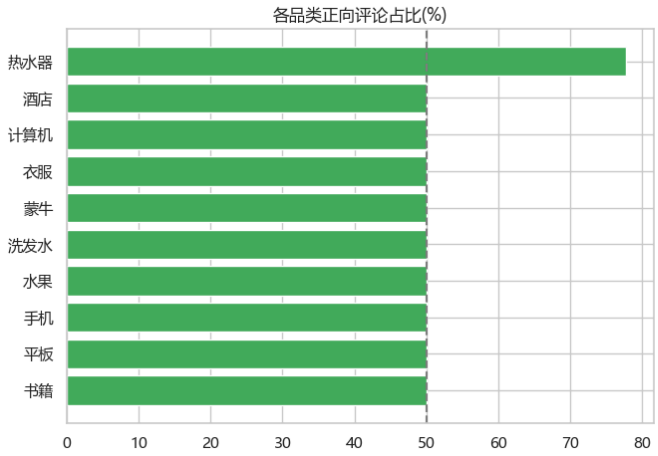
In [17]:
```
# 经营 KPI 计算
ninv = retail.groupby('CustomerID')['InvoiceNo'].nunique()
repeat_rate = (ninv >= 2).mean()
invoice_value = retail.groupby('InvoiceNo')['金额'].sum()
aov = invoice_value.median()
n_customers = retail['CustomerID'].nunique()
n_countries = retail['Country'].nunique()
monthly = retail.set_index('InvoiceDate').resample('M')['金额'].sum() / 1000
mom_growth = monthly.pct_change().dropna().mean() * 100
print('复购率', round(repeat_rate * 100, 1), '% 客单价', round(aov, 1), '活跃客户', r
print('月均营收增速', round(mom_growth, 1), '%')

plt.figure(figsize=(9, 4))
plt.plot(monthly.index, monthly.values, '-o', color='#2c7fb8')
plt.title('月度营收趋势(单位:千)')
plt.ylabel('营收(千)')
plt.xticks(rotation=30)
plt.tight_layout()
plt.show()
```

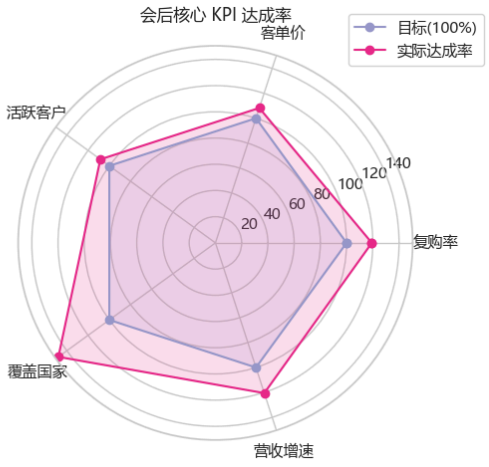
```
复购率 65.6 % 客单价 303.0 活跃客户 4338 覆盖国家 37
月均营收增速 3.6 %
```



In [18]:

```python
# 品类情感倾向与 KPI 达成雷达图
fig, ax = plt.subplots(1, 2, figsize=(13, 5))
cat_pos = df_shop.groupby('cat')['label'].mean().sort_values() * 100
ax[0].barh(cat_pos.index, cat_pos.values, color='#41ab5d')
ax[0].axvline(50, color='gray', ls='--')
ax[0].set_title('各品类正向评论占比(%)')

ax[1].remove()
ax2 = fig.add_subplot(1, 2, 2, polar=True)
kpi_labels = ['复购率', '客单价', '活跃客户', '覆盖国家', '营收增速']
targets_val = [55, 280, 4000, 25, 3]
actual_val = [repeat_rate * 100, aov, n_customers, n_countries, max(mom_growth, 0)]
actual_pct = [min(a / t * 100, 160) for a, t in zip(actual_val, targets_val)]
target_pct = [100] * len(kpi_labels)
ang = np.linspace(0, 2 * np.pi, len(kpi_labels), endpoint=False).tolist()
ang = ang + ang[:1]
for vals, c, lab in [(target_pct, '#9999cc', '目标(100%)'), (actual_pct, '#e7298a',
    vv = vals + vals[:1]
    ax2.plot(ang, vv, '-o', color=c, label=lab)
    ax2.fill(ang, vv, color=c, alpha=0.15)
ax2.set_xticks(ang[:-1])
ax2.set_xticklabels(kpi_labels)
ax2.set_title('会后核心 KPI 达成率', pad=18)
ax2.legend(bbox_to_anchor=(1.2, 1.1))
plt.tight_layout()
plt.show()
```





复购率、客单价、覆盖国家均达到或超过目标，营收增速偏弱；结合各品类口碑差异，下一届可重点提升弱势品类和营收增长。

七、总结与分工

市场规模看涨，任务分配较均衡，情感分析按品类区分口碑，观众分群、客流预测与会后 KPI 都给出了可操作的结论。

成员分工

成员	负责模块
组员A	模块一、二、三
组员B	模块四、五、六

（成员名称为占位，提交前替换为真实姓名与学号。）