In [88]:
```
import pandas as pd
```

In [89]:
```
data=pd.read_csv('广东.csv')
data
```

Out[89]:

	省份	时间	累计确诊	新增确诊	现有确诊	累计治愈	新增治愈	累计死亡	新增死亡	新增无症状	新增本土确诊
0	广东	2022/12/20	62367	1189	10000	52359	993	8	0	0	1171
1	广东	2022/12/19	61178	1111	9804	51366	917	8	0	0	1075
2	广东	2022/12/18	60067	502	9610	50449	568	8	0	0	846
3	广东	2022/12/17	59565	932	9676	49881	789	8	0	0	915
4	广东	2022/12/16	58633	1003	9533	49092	837	8	0	0	990
...	...	...	...	...	...	...	...	...	...	...	...
1062	广东	2020/1/23	53	21	51	2	2	0	0	0	0
1063	广东	2020/1/22	32	6	32	0	0	0	0	0	0
1064	广东	2020/1/21	26	12	26	0	0	0	0	0	0
1065	广东	2020/1/20	14	13	14	0	0	0	0	0	0
1066	广东	2020/1/19	1	0	1	0	0	0	0	0	0

1067 rows × 11 columns

查询缺失值

In [90]:
```
data.isnull().sum()
```

Out[90]:
```
省份        0
时间        0
累计确诊      0
新增确诊      0
现有确诊      0
累计治愈      0
新增治愈      0
累计死亡      0
新增死亡      0
新增无症状     0
新增本土确诊    0
dtype: int64
```

In [91]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1067 entries, 0 to 1066
Data columns (total 11 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   省份      1067 non-null   object
 1   时间      1067 non-null   object
 2   累计确诊    1067 non-null   int64 
 3   新增确诊    1067 non-null   int64 
 4   现有确诊    1067 non-null   int64 
 5   累计治愈    1067 non-null   int64 
 6   新增治愈    1067 non-null   int64 
 7   累计死亡    1067 non-null   int64 
 8   新增死亡    1067 non-null   int64 
 9   新增无症状   1067 non-null   int64 
 10  新增本土确诊  1067 non-null   int64 
dtypes: int64(9), object(2)
memory usage: 91.8+ KB
```

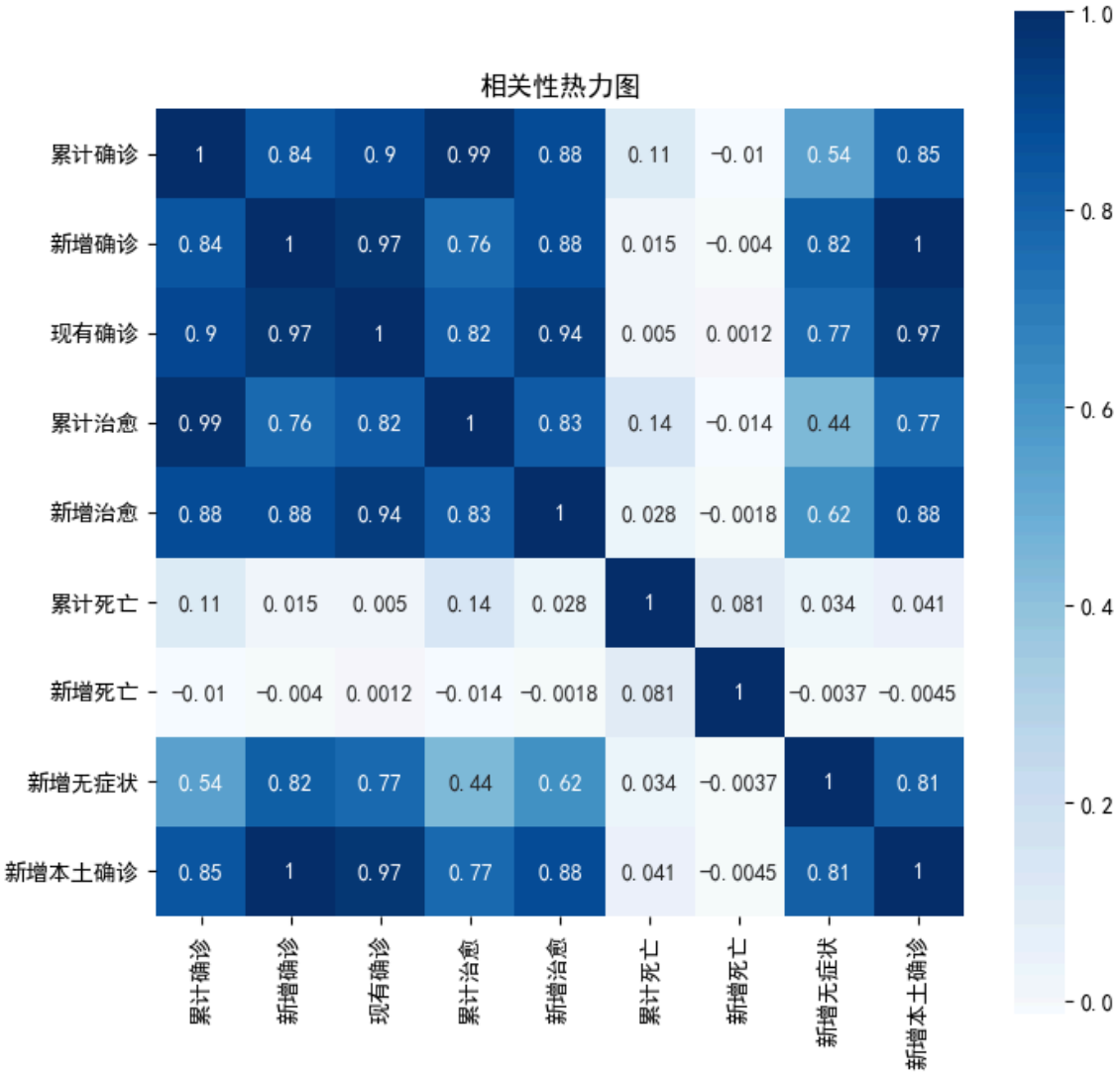
相关性

In [92]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# 计算相关系数矩阵，包含了任意两个菜品间的相关系数
print('相关系数矩阵为：\n', data.corr())

# 绘制相关性热力图
plt.subplots(figsize=(8, 8))  # 设置画面大小
plt.rcParams['font.sans-serif'] = ['SimHei']  # 用来正常显示中文标签
plt.rcParams['axes.unicode_minus'] = False  # 用来正常显示负号
sns.heatmap(data.corr(), annot=True, vmax=1, square=True, cmap="Blues")
plt.title('相关性热力图')
plt.show()
```

相关系数矩阵为：

	累计确诊	新增确诊	现有确诊	累计治愈	新增治愈	累计死亡	新增死亡
累计确诊	1.000000	0.842339	0.898650	0.989205	0.884716	0.106450	-0.010224
新增确诊	0.842339	1.000000	0.966543	0.763231	0.882690	0.014898	-0.003956
现有确诊	0.898650	0.966543	1.000000	0.824666	0.941912	0.005017	0.001224
累计治愈	0.989205	0.763231	0.824666	1.000000	0.826104	0.135385	-0.013609
新增治愈	0.884716	0.882690	0.941912	0.826104	1.000000	0.027594	-0.001813
累计死亡	0.106450	0.014898	0.005017	0.135385	0.027594	1.000000	0.080782
新增死亡	-0.010224	-0.003956	0.001224	-0.013609	-0.001813	0.080782	1.000000
新增无症状	0.542284	0.816666	0.773388	0.440844	0.620285	0.033530	-0.003673
新增本土确诊	0.847004	0.997561	0.966654	0.769203	0.883657	0.041256	-0.004520

	新增无症状	新增本土确诊
累计确诊	0.542284	0.847004
新增确诊	0.816666	0.997561
现有确诊	0.773388	0.966654
累计治愈	0.440844	0.769203
新增治愈	0.620285	0.883657
累计死亡	0.033530	0.041256
新增死亡	-0.003673	-0.004520
新增无症状	1.000000	0.813444
新增本土确诊	0.813444	1.000000


相关性热力图
累计确诊 1 0.84 0.9 0.99 0.88 0.11 -0.01 0.54 0.85
新增确诊 0.84 1 0.97 0.76 0.88 0.015 -0.004 0.82 1
现有确诊 0.9 0.97 1 0.82 0.94 0.005 0.0012 0.77 0.97
累计治愈 0.99 0.76 0.82 1 0.83 0.14 -0.014 0.44 0.77
新增治愈 0.88 0.88 0.94 0.83 1 0.028 -0.0018 0.62 0.88
累计死亡 0.11 0.015 0.005 0.14 0.028 1 0.081 0.034 0.041
新增死亡 -0.01 -0.004 0.0012 -0.014 -0.0018 0.081 1 -0.0037 -0.0045
新增无症状 0.54 0.82 0.77 0.44 0.62 0.034 -0.0037 1 0.81
新增本土确诊 0.85 1 0.97 0.77 0.88 0.041 -0.0045 0.81 1
累计确诊
新增确诊
现有确诊
累计治愈
新增治愈
累计死亡
新增死亡
新增无症状
新增本土确诊
1.0
0.8
0.6
0.4
0.2
0.0

In [93]:
```
data
```

Out[93]:

	省份	时间	累计确诊	新增确诊	现有确诊	累计治愈	新增治愈	累计死亡	新增死亡	新增无症状	新增本土确诊
0	广东	2022/12/20	62367	1189	10000	52359	993	8	0	0	1171
1	广东	2022/12/19	61178	1111	9804	51366	917	8	0	0	1075
2	广东	2022/12/18	60067	502	9610	50449	568	8	0	0	846
3	广东	2022/12/17	59565	932	9676	49881	789	8	0	0	915
4	广东	2022/12/16	58633	1003	9533	49092	837	8	0	0	990
...	...	...	...	...	...	...	...	...	...	...	...
1062	广东	2020/1/23	53	21	51	2	2	0	0	0	0
1063	广东	2020/1/22	32	6	32	0	0	0	0	0	0
1064	广东	2020/1/21	26	12	26	0	0	0	0	0	0
1065	广东	2020/1/20	14	13	14	0	0	0	0	0	0
1066	广东	2020/1/19	1	0	1	0	0	0	0	0	0

1067 rows × 11 columns

回归

In [94]:
```
import numpy as np
import pandas as pd
import statsmodels.api as sm
```

In [95]:
```
data2=data.drop(labels=['时间'],axis=1)
```

In [96]:
```
data2
```

Out[96]:

	省份	累计确诊	新增确诊	现有确诊	累计治愈	新增治愈	累计死亡	新增死亡	新增无症状	新增本土确诊
0	广东	62367	1189	10000	52359	993	8	0	0	1171
1	广东	61178	1111	9804	51366	917	8	0	0	1075
2	广东	60067	502	9610	50449	568	8	0	0	846
3	广东	59565	932	9676	49881	789	8	0	0	915
4	广东	58633	1003	9533	49092	837	8	0	0	990
...	...	...	...	...	...	...	...	...	...	...
1062	广东	53	21	51	2	2	0	0	0	0
1063	广东	32	6	32	0	0	0	0	0	0
1064	广东	26	12	26	0	0	0	0	0	0
1065	广东	14	13	14	0	0	0	0	0	0
1066	广东	1	0	1	0	0	0	0	0	0

1067 rows × 10 columns

In [97]:
```
data2.to_csv('./new.csv')
```

累计治愈

In [98]:
```
data3=pd.read_csv('./累计治愈.csv')
```

In [99]:
```
data3
```

Out[99]:

	新增确诊	现有确诊	新增治愈	累计死亡	新增死亡	新增无症状	新增本土确诊	累计治愈
0	1189	10000	993	8	0	0	1171	52359
1	1111	9804	917	8	0	0	1075	51366
2	502	9610	568	8	0	0	846	50449
3	932	9676	789	8	0	0	915	49881
4	1003	9533	837	8	0	0	990	49092
...	...	...	...	...	...	...	...	...
1062	21	51	2	0	0	0	0	2
1063	6	32	0	0	0	0	0	0
1064	12	26	0	0	0	0	0	0
1065	13	14	0	0	0	0	0	0
1066	0	1	0	0	0	0	0	0

1067 rows × 8 columns

In [100]:
```
# 分割自变量和目标变量
X = data3.iloc[:, :-1]
y = data3['累计治愈']
```

In [101]:
```
from sklearn.model_selection import train_test_split
# 划分训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

In [102]:
```
from sklearn.ensemble import RandomForestRegressor
# 训练模型
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
```

Out[102]:
```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

In [103]:
```
# 预测结果
y_pred = rf.predict(X_test)
```

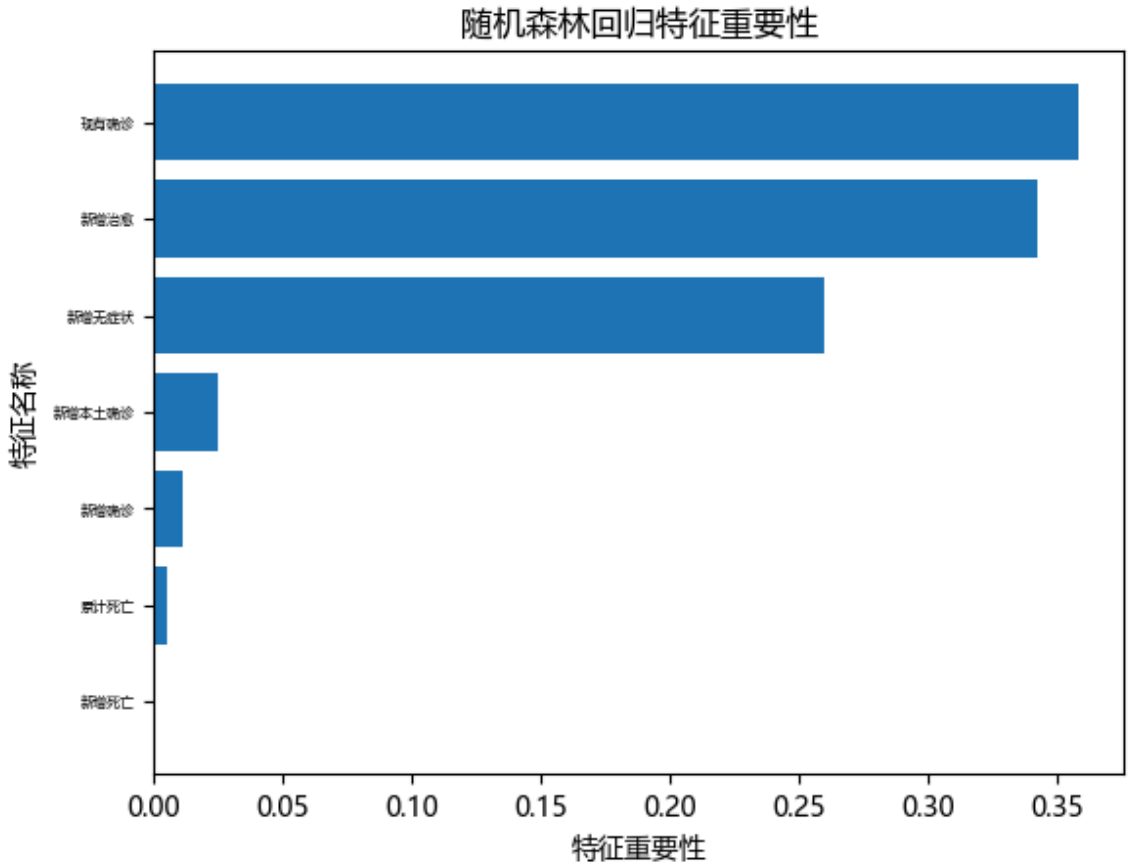
In [104]:
```
from sklearn.metrics import mean_squared_error, r2_score
# 计算MSE和R-squared
r2 = r2_score(y_test, y_pred)

# 输出模型评估结果和目标方程
print('R-squared:', r2)
```

```
R-squared: 0.9604504754779675
```

In [105]:
```
# 输出目标方程
print("目标方程：")
for i, feature in enumerate(X.columns):
    print("{} * {} +".format(rf.feature_importances_[i], feature), end=' ')
```

目标方程：
0.011085749154049744 * 新增确诊 + 0.3577627862949939 * 现有确诊 + 0.34191208968005576 * 新增治愈 + 0.004995604265516014 * 累计死亡 + 9.427960699552539e-05 * 新增死亡 + 0.25958005230778136 * 新增无症状 + 0.02456943869060767 * 新增本土确诊 +

In [106]:
```
import matplotlib.pyplot as plt
# 绘制特征重要性条形图
feature_importance = rf.feature_importances_
feature_names = X.columns.tolist()
sorted_idx = feature_importance.argsort()
#避免中文乱码
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']
plt.rcParams['axes.unicode_minus'] = False
plt.barh(range(len(feature_importance)), feature_importance[sorted_idx])
plt.yticks(range(len(feature_importance)), [feature_names[i] for i in sorted_idx],fo
plt.xlabel('特征重要性')
plt.ylabel('特征名称')
plt.title('随机森林回归特征重要性')
plt.savefig('随机森林回归特征重要性',dpi=300)
```



累计确诊

In [107]:
```
data4=pd.read_csv('./累计确诊.csv')
```

In [108]:
```
data4
```

Out[108]:

	新增确诊	现有确诊	新增治愈	累计死亡	新增死亡	新增无症状	新增本土确诊	累计确诊
0	1189	10000	993	8	0	0	1171	62367
1	1111	9804	917	8	0	0	1075	61178
2	502	9610	568	8	0	0	846	60067
3	932	9676	789	8	0	0	915	59565
4	1003	9533	837	8	0	0	990	58633
...	...	...	...	...	...	...	...	...
1062	21	51	2	0	0	0	0	53
1063	6	32	0	0	0	0	0	32
1064	12	26	0	0	0	0	0	26
1065	13	14	0	0	0	0	0	14
1066	0	1	0	0	0	0	0	1

1067 rows × 8 columns

In [116]:
```
# 分割自变量和目标变量
X = data4.iloc[:, :-1]
y = data4['累计确诊']
```

In [117]:
```
from sklearn.model_selection import train_test_split
# 划分训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

In [118]:
```
from sklearn.ensemble import RandomForestRegressor
# 训练模型
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
```

Out[118]:

RandomForestRegressor

RandomForestRegressor(random_state=42)

In [119]:
```
# 预测结果
y_pred = rf.predict(X_test)
```

In [120]:
```
from sklearn.metrics import mean_squared_error, r2_score
# 计算MSE和R-squared
r2 = r2_score(y_test, y_pred)

# 输出模型评估结果和目标方程
print('R-squared:', r2)
```

```
R-squared: 0.9828006956630415
```

In [121]:
```
# 输出目标方程
print("目标方程：")
for i, feature in enumerate(X.columns):
    print("{} * {} +".format(rf.feature_importances_[i], feature), end=' ')
```

目标方程：
0.02235413301736211 * 新增确诊 + 0.3957421883334026 * 现有确诊 + 0.366842918588497
44 * 新增治愈 + 0.002112254274132873 * 累计死亡 + 1.1751815598148054e-05 * 新增死
亡 + 0.1727301736993944 * 新增无症状 + 0.04020658027161243 * 新增本土确诊 +

In [122]:
```
import matplotlib.pyplot as plt
# 绘制特征重要性条形图
feature_importance = rf.feature_importances_
feature_names = X.columns.tolist()
sorted_idx = feature_importance.argsort()
#避免中文乱码
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']
plt.rcParams['axes.unicode_minus'] = False
plt.barh(range(len(feature_importance)), feature_importance[sorted_idx])
plt.yticks(range(len(feature_importance)), [feature_names[i] for i in sorted_idx],fo
plt.xlabel('特征重要性')
plt.ylabel('特征名称')
plt.title('随机森林回归特征重要性')
plt.savefig('随机森林回归特征重要性',dpi=300)
```