GA Configuration results

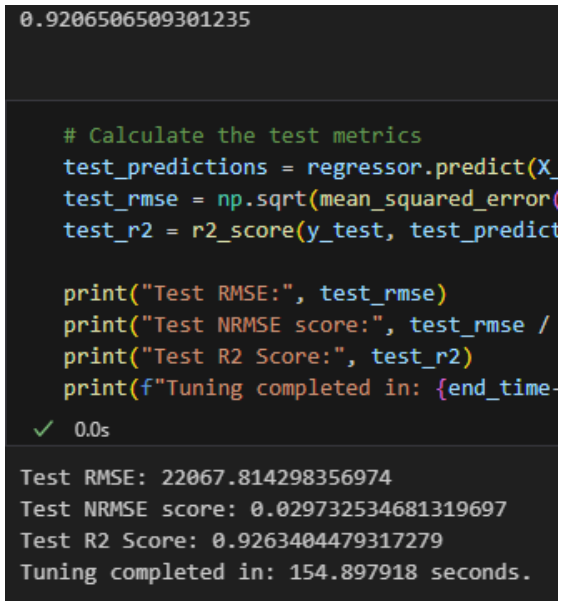
In this document the results of the GA configuration analysis are included.

Below the results of the models and the respective fitness histories for the GA are shown.

The tested parameters are:
- Crossover probability
- Population size
- General mutation probability
- Global mutation probability
- Local mutation probability
- Number of generations

The tested ranges can be found in the respective sections.

Mean CV Score (measured by R^2)



Runtime for the GA

Crossover probability : 0.1

```
0.9188753574525774
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X
test_rmse = np.sqrt(mean_squared_error
test_r2 = r2_score(y_test, test_predic

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time
```

✓ 0.0s

```
Test RMSE: 20371.17389929225
Test NRMSE score: 0.027446607365415293
Test R2 Score: 0.9372313852172687
Tuning completed in: 140.197024 seconds.
```

Crossover probability : 0.3

```
0.9215489917210066
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X_tes
test_rmse = np.sqrt(mean_squared_error(y_t
test_r2 = r2_score(y_test, test_prediction

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse / (y.
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time-sta
```

✓ 0.0s

```
Test RMSE: 21336.294125228742
Test NRMSE score: 0.028746938707764694
Test R2 Score: 0.9311429502037037
Tuning completed in: 155.319988 seconds.
```

Crossover probability : 0.5

```
0.9194947122991879
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X
test_rmse = np.sqrt(mean_squared_error
test_r2 = r2_score(y_test, test_predic

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time
```

✓ 0.0s

```
Test RMSE: 20343.723097416452
Test NRMSE score: 0.02740962219290263
Test R2 Score: 0.937400436631342
Tuning completed in: 131.688353 seconds.
```

Crossover probability : 0.7

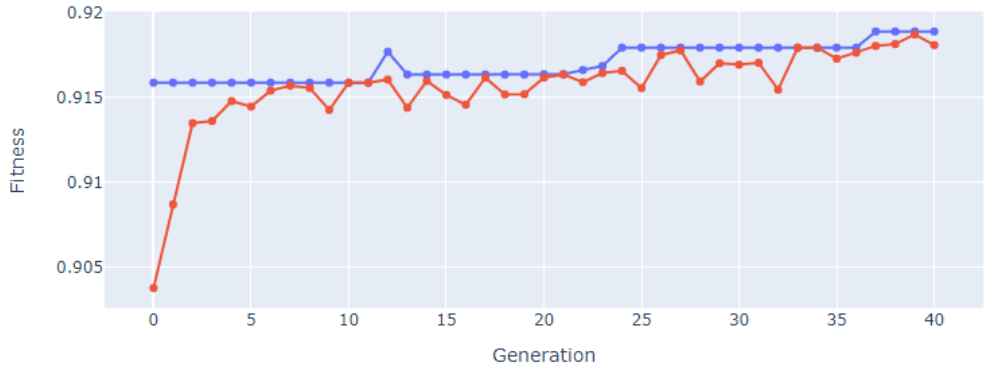
```
0.9169569309420236
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X
test_rmse = np.sqrt(mean_squared_error
test_r2 = r2_score(y_test, test_predic

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time
```

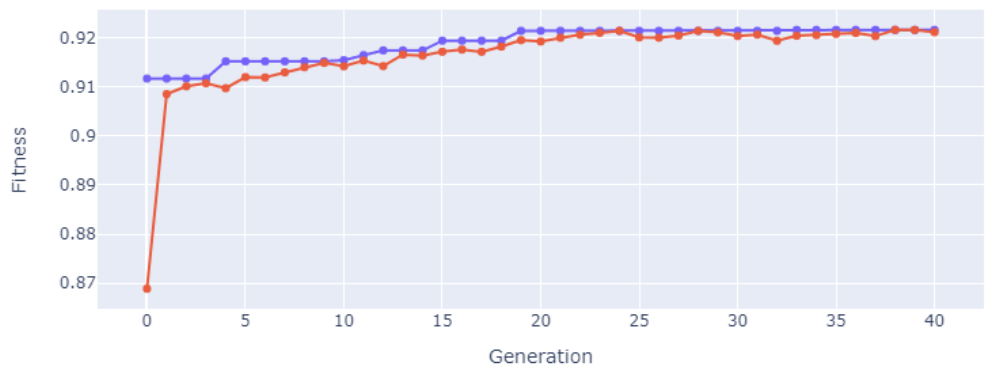
✓ 0.0s

```
Test RMSE: 20691.790049884854
Test NRMSE score: 0.027878581764329623
Test R2 Score: 0.9352400419892514
Tuning completed in: 175.047021 seconds.
```

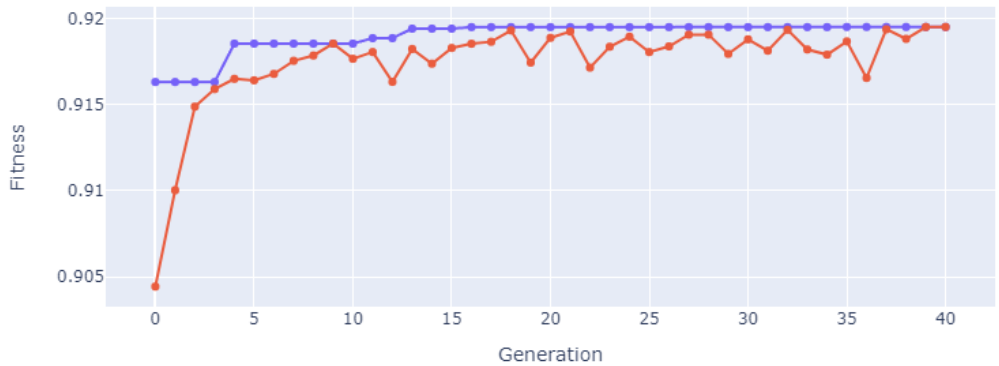
Fitness history per generation - Crossover probability = 0.1



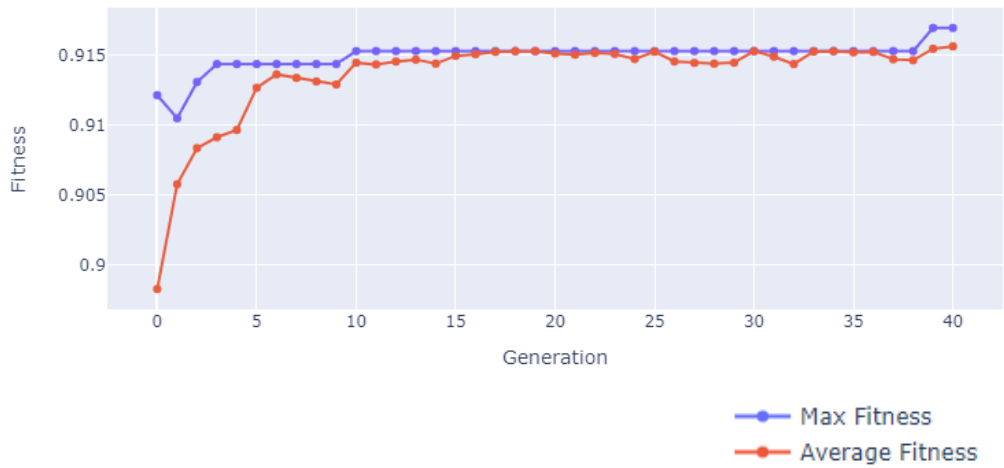
Fitness history per generation - Crossover probability = 0.3



Fitness history per generation - Crossover probability = 0.5



Fitness history per generation - Crossover probability = 0.7



Population size : 5

```
0.9206506509301235
```

```
    # Calculate the test metrics
    test_predictions = regressor.predict(X_
    test_rmse = np.sqrt(mean_squared_error(
    test_r2 = r2_score(y_test, test_predict

    print("Test RMSE:", test_rmse)
    print("Test NRMSE score:", test_rmse /
    print("Test R2 Score:", test_r2)
    print(f"Tuning completed in: {end_time-
```

✓ 0.0s

```
Test RMSE: 22067.814298356974
Test NRMSE score: 0.029732534681319697
Test R2 Score: 0.9263404479317279
Tuning completed in: 154.897918 seconds.
```

Population size : 15

```
0.9196068954337002
```

```
    # Calculate the test metrics
    test_predictions = regressor.predict(X_t
    test_rmse = np.sqrt(mean_squared_error(y
    test_r2 = r2_score(y_test, test_predicti

    print("Test RMSE:", test_rmse)
    print("Test NRMSE score:", test_rmse / (
    print("Test R2 Score:", test_r2)
    print(f"Tuning completed in: {end_time-s
```

✓ 0.0s

```
Test RMSE: 21750.042984890264
Test NRMSE score: 0.02930439320474941
Test R2 Score: 0.928446534690796
Tuning completed in: 541.843413 seconds.
```

Population size : 20

```
0.9201608187862899
```

```
    # Calculate the test metrics
    test_predictions = regressor.predict(X
    test_rmse = np.sqrt(mean_squared_error
    test_r2 = r2_score(y_test, test_predic

    print("Test RMSE:", test_rmse)
    print("Test NRMSE score:", test_rmse /
    print("Test R2 Score:", test_r2)
    print(f"Tuning completed in: {end_time
```

✓ 0.0s

```
Test RMSE: 20754.986599653574
Test NRMSE score: 0.027963728103805485
Test R2 Score: 0.9348438601522177
Tuning completed in: 650.119274 seconds.
```

Population size : 25

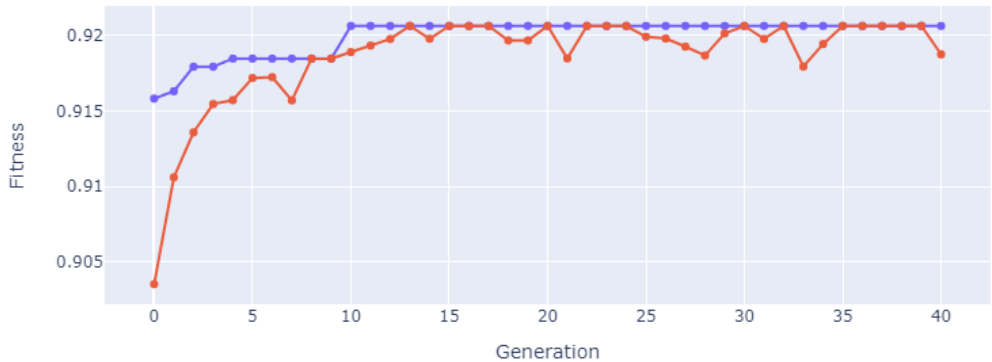
```
0.9230350190410579
```

```
    # Calculate the test metrics
    test_predictions = regressor.predict(X_tes
    test_rmse = np.sqrt(mean_squared_error(y_t
    test_r2 = r2_score(y_test, test_predictior

    print("Test RMSE:", test_rmse)
    print("Test NRMSE score:", test_rmse / (y.
    print("Test R2 Score:", test_r2)
    print(f"Tuning completed in: {end_time-sta
```

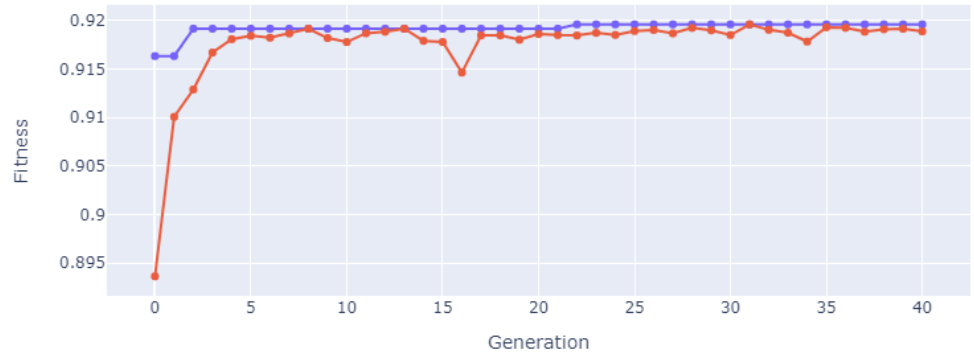
✓ 0.1s

```
Test RMSE: 19982.085779988778
Test NRMSE score: 0.0269223789191871
Test R2 Score: 0.9396062398122474
Tuning completed in: 1313.378993 seconds.
```

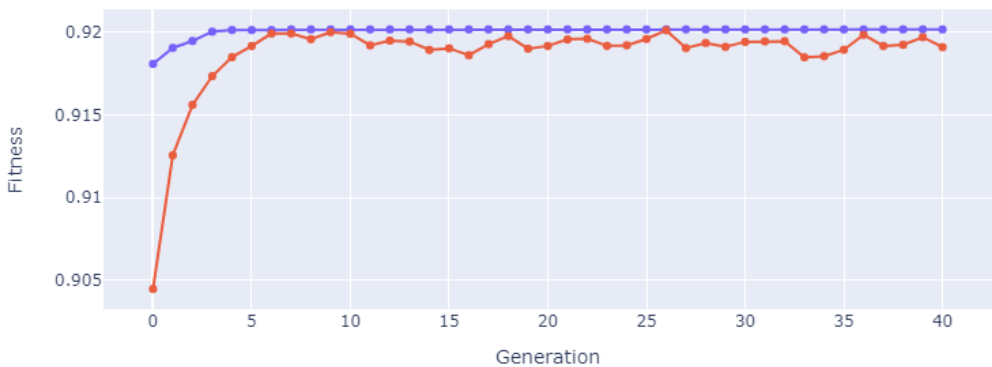
Fitness history per generation - Population size = 5



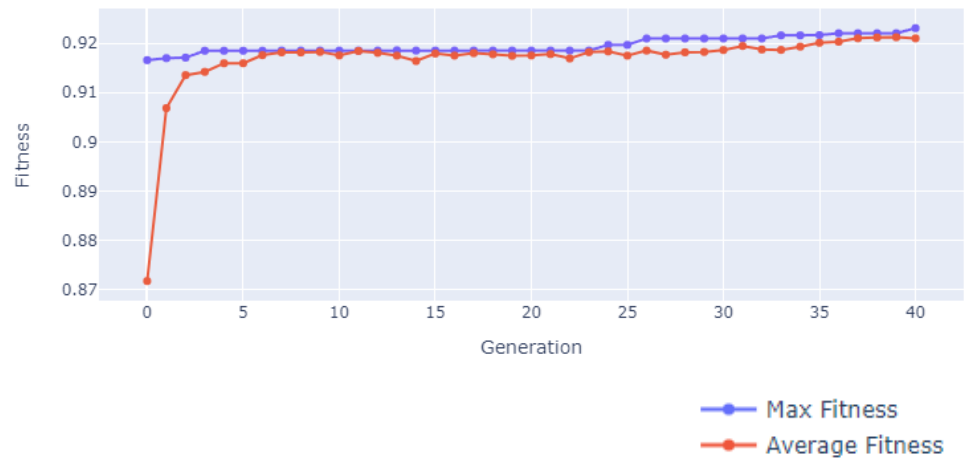
Fitness history per generation - Population size = 15



Fitness history per generation - Population size = 20



Fitness history per generation - Population size = 25



General mutation probability : 0.1

```
0.9224524077742459
```

```
# Calculate the test metrics
test_predictions = regressor.predict(
test_rmse = np.sqrt(mean_squared_erro
test_r2 = r2_score(y_test, test_predi

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_tim
```

✓ 0.0s

```
Test RMSE: 20278.815302370374
Test NRMSE score: 0.027322170248582106
Test R2 Score: 0.9377992542593621
Tuning completed in: 288.270372 seconds.
```

General mutation probability : 0.3

```
0.9201193506689254
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X
test_rmse = np.sqrt(mean_squared_error
test_r2 = r2_score(y_test, test_predic

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time
```

✓ 0.0s

```
Test RMSE: 20590.252531095866
Test NRMSE score: 0.02774177764961159
Test R2 Score: 0.9358740549795291
Tuning completed in: 231.000848 seconds.
```

General mutation probability : 0.5

```
0.9207377424226978
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X_
test_rmse = np.sqrt(mean_squared_error
test_r2 = r2_score(y_test, test_predict

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time-
```

✓ 0.0s

```
Test RMSE: 20320.83728361718
Test NRMSE score: 0.027378787546421677
Test R2 Score: 0.9375412010444256
Tuning completed in: 297.252692 seconds.
```

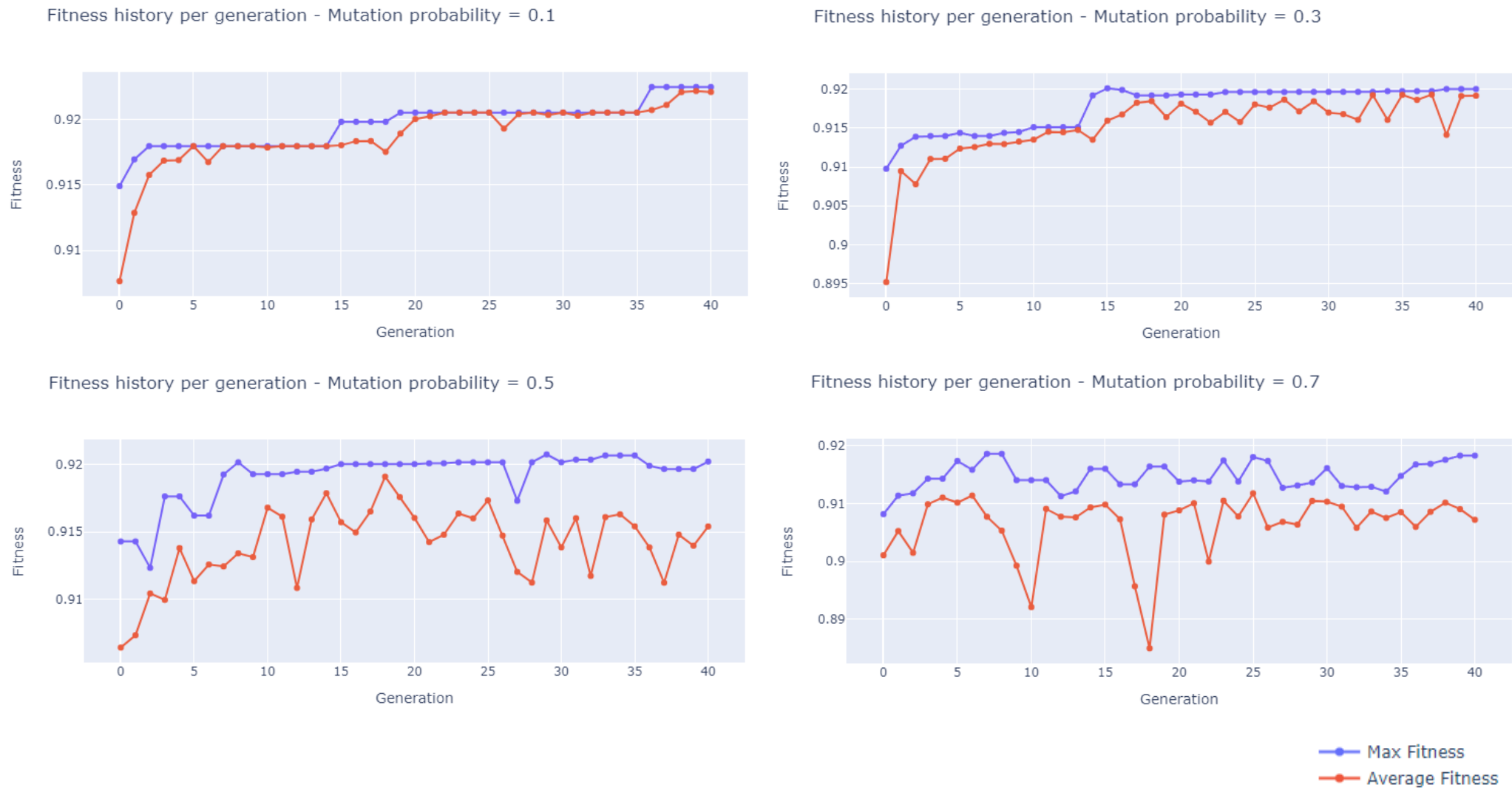
General mutation probability : 0.7

```
0.9186004577207054
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X_t
test_rmse = np.sqrt(mean_squared_error(y_
test_r2 = r2_score(y_test, test_predicti

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse / (y
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time-st
```

✓ 0.0s

```
Test RMSE: 20785.338970268855
Test NRMSE score: 0.028004622634626615
Test R2 Score: 0.9346531503812373
Tuning completed in: 309.502645 seconds.
```


Fitness history per generation - Mutation probability = 0.1
Fitness
Generation
Fitness history per generation - Mutation probability = 0.3
Fitness
Generation
Fitness history per generation - Mutation probability = 0.5
Fitness
Generation
Fitness history per generation - Mutation probability = 0.7
Fitness
Generation
Max Fitness
Average Fitness

Global mutation probability : 0.1

```
0.9215049888977825
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X_
test_rmse = np.sqrt(mean_squared_error(
test_r2 = r2_score(y_test, test_predict

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time-
```

✓ 0.0s

```
Test RMSE: 20740.304776615532
Test NRMSE score: 0.027943946905415754
Test R2 Score: 0.934936008857116
Tuning completed in: 296.636440 seconds.
```

Global mutation probability : 0.3

```
0.9211325897551733
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X
test_rmse = np.sqrt(mean_squared_error
test_r2 = r2_score(y_test, test_predic

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time
```

✓ 0.0s

```
Test RMSE: 19980.301170050894
Test NRMSE score: 0.02691997446824541
Test R2 Score: 0.9396170269235523
Tuning completed in: 320.406416 seconds.
```

Global mutation probability : 0.5

```
0.9199887079737185
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X_
test_rmse = np.sqrt(mean_squared_error(
test_r2 = r2_score(y_test, test_predict

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time-
```

✓ 0.0s

```
Test RMSE: 20220.97034584849
Test NRMSE score: 0.027244234248547234
Test R2 Score: 0.9381536011741138
Tuning completed in: 391.320688 seconds.
```

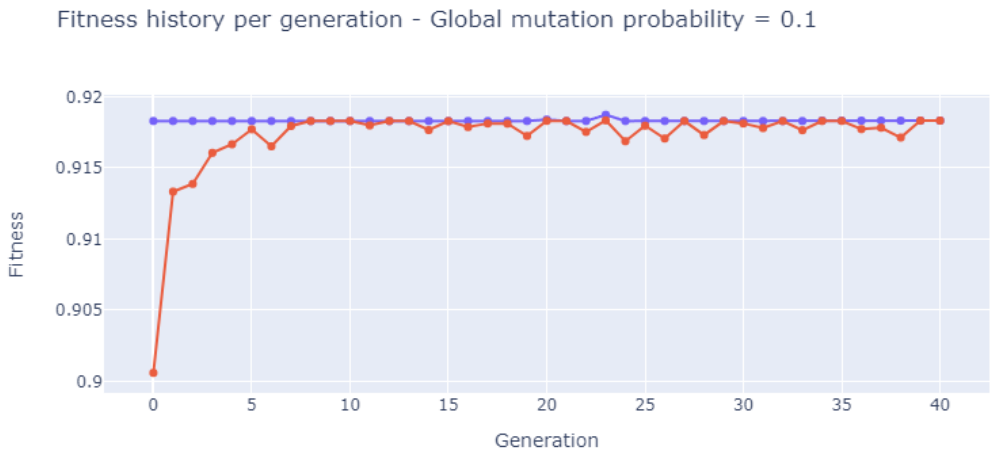
Global mutation probability : 0.7

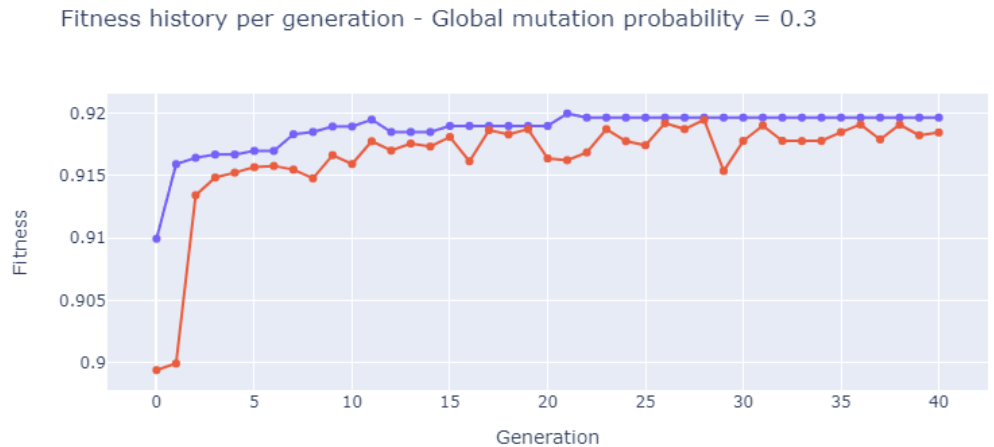
```
0.9187220267270988
```

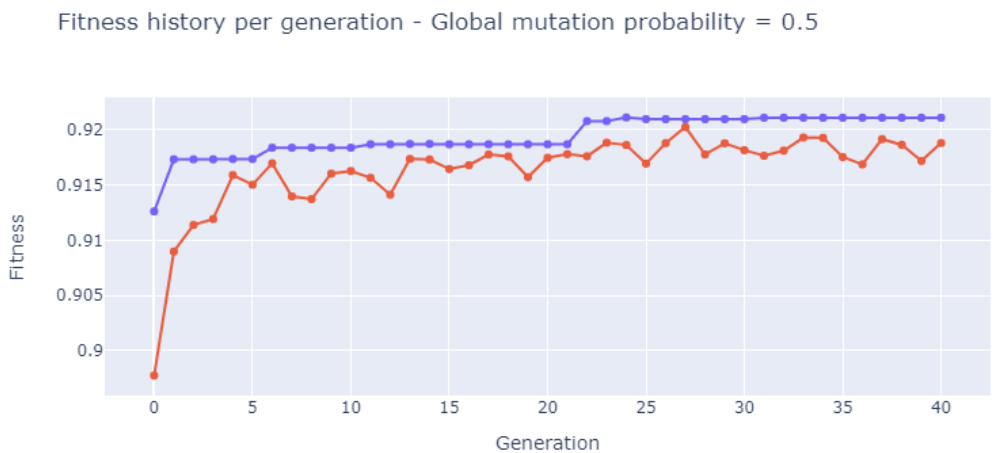
```
# Calculate the test metrics
test_predictions = regressor.predict(
test_rmse = np.sqrt(mean_squared_erro
test_r2 = r2_score(y_test, test_predi

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_tim
```

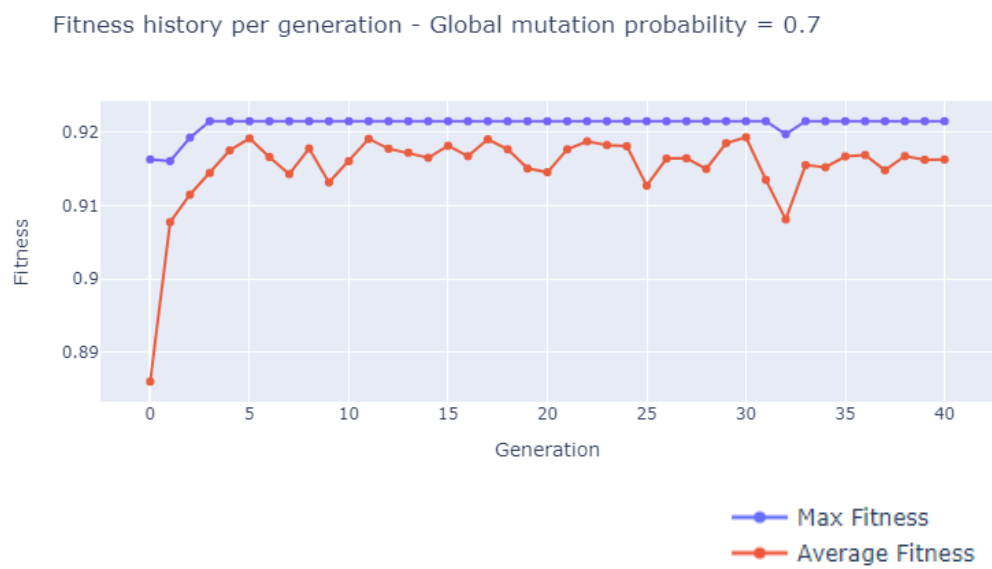
✓ 0.0s

```
Test RMSE: 20529.556933585613
Test NRMSE score: 0.02766000090753925
Test R2 Score: 0.9362515565101833
Tuning completed in: 591.687601 seconds.
```


Fitness history per generation - Global mutation probability = 0.1
Fitness
Generation
0.92
0.915
0.91
0.905
0.9
0
5
10
15
20
25
30
35
40


Fitness history per generation - Global mutation probability = 0.3
Fitness
Generation
0.92
0.915
0.91
0.905
0.9
0
5
10
15
20
25
30
35
40


Fitness history per generation - Global mutation probability = 0.5
Fitness
Generation
0.92
0.915
0.91
0.905
0.9
0
5
10
15
20
25
30
35
40


Fitness history per generation - Global mutation probability = 0.7
Fitness
Generation
0.92
0.91
0.9
0.89
0
5
10
15
20
25
30
35
40
Max Fitness
Average Fitness

Local mutation probability : 0.1

```
0.9209827188370546
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X_
test_rmse = np.sqrt(mean_squared_error(
test_r2 = r2_score(y_test, test_predict

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time-
```

✓ 0.0s

```
Test RMSE: 19887.44482088421
Test NRMSE score: 0.026794866716990468
Test R2 Score: 0.9401769697939176
Tuning completed in: 281.739142 seconds.
```

Local mutation probability : 0.3

```
0.9212001325240408
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X_
test_rmse = np.sqrt(mean_squared_error(
test_r2 = r2_score(y_test, test_predict

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time-
```

✓ 0.0s

```
Test RMSE: 20140.54515414269
Test NRMSE score: 0.02713587531597172
Test R2 Score: 0.9386445881836268
Tuning completed in: 341.166647 seconds.
```

Local mutation probability : 0.5

```
0.9189424622069335
```

```
# Calculate the test metrics
test_predictions = regressor.predict(
test_rmse = np.sqrt(mean_squared_erro
test_r2 = r2_score(y_test, test_predi

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_tim
```

✓ 0.0s

```
Test RMSE: 20371.24400864133
Test NRMSE score: 0.027446701825547357
Test R2 Score: 0.9372309531681075
Tuning completed in: 287.847100 seconds.
```

Local mutation probability : 0.7

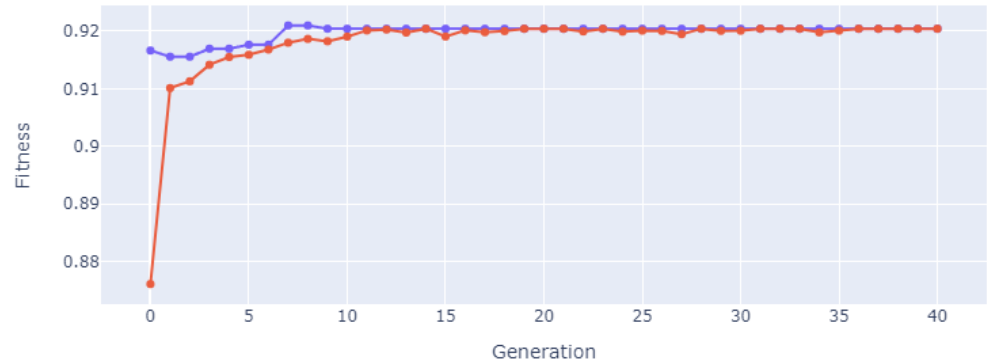
```
0.9212739462233227
```

```
# Calculate the test metrics
test_predictions = regressor.predict(X_
test_rmse = np.sqrt(mean_squared_error
test_r2 = r2_score(y_test, test_predict

print("Test RMSE:", test_rmse)
print("Test NRMSE score:", test_rmse /
print("Test R2 Score:", test_r2)
print(f"Tuning completed in: {end_time
```

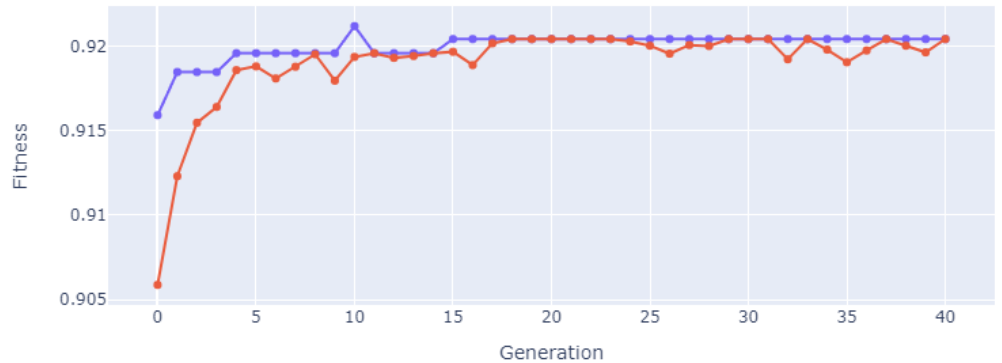
✓ 0.0s

```
Test RMSE: 19799.706039904497
Test NRMSE score: 0.02667665399718476
Test R2 Score: 0.9407036560072677
Tuning completed in: 278.862958 seconds.
```

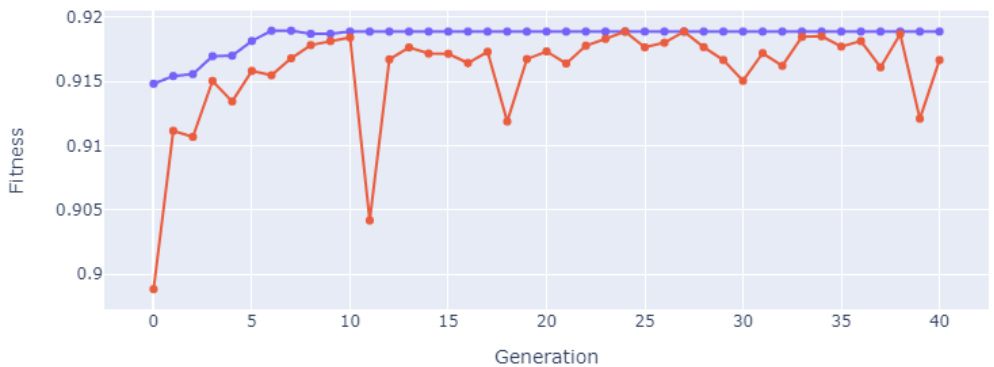
Fitness history per generation - Local mutation probability = 0.1



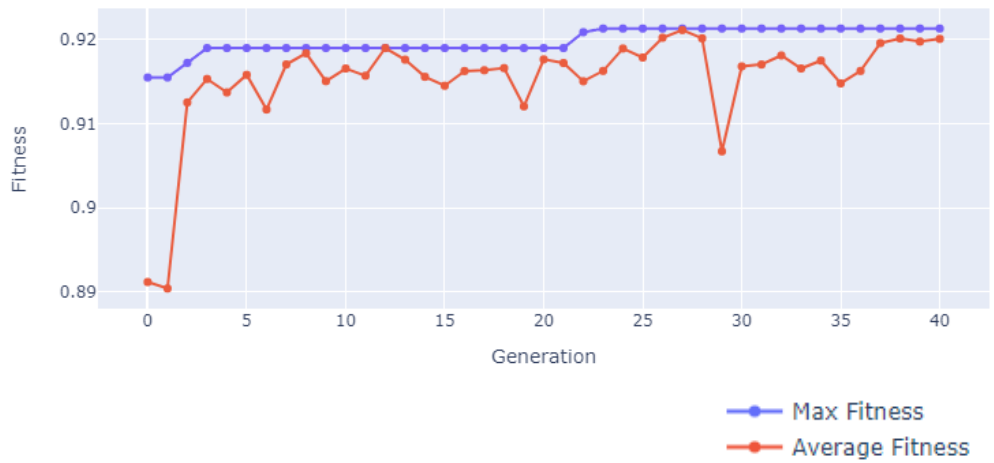
Fitness history per generation - Local mutation probability = 0.3



Fitness history per generation - Local mutation probability = 0.5



Fitness history per generation - Local mutation probability = 0.7



Number of generations: 10

```
0.918455216165848

    # Calculate the test metrics
    test_predictions = regressor.predict(
    test_rmse = np.sqrt(mean_squared_erro
    test_r2 = r2_score(y_test, test_predi

    print("Test RMSE:", test_rmse)
    print("Test NRMSE score:", test_rmse /
    print("Test R2 Score:", test_r2)
    print(f"Tuning completed in: {end_tim
 ✓ 0.0s

Test RMSE: 20384.687405866596
Test NRMSE score: 0.027464814460937115
Test R2 Score: 0.9371480806975142
Tuning completed in: 133.334631 seconds.
```

Number of generations: 20

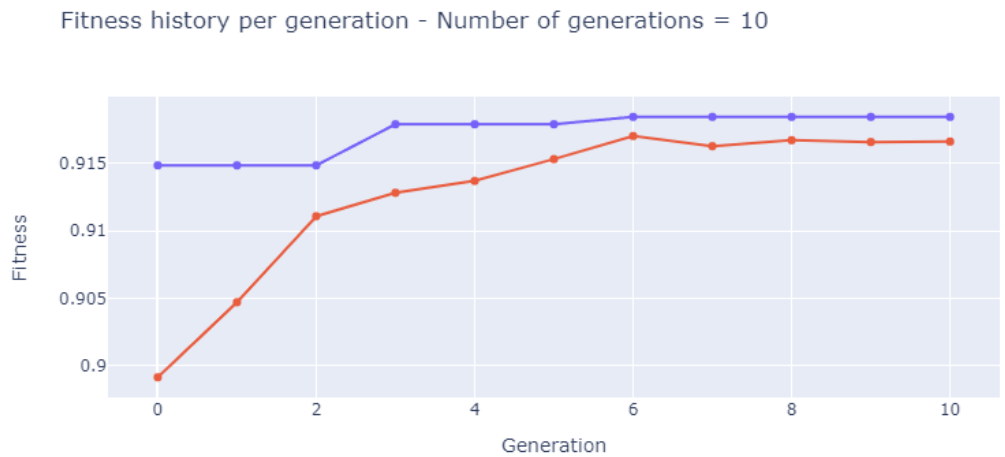
```
0.9207361377479716

    # Calculate the test metrics
    test_predictions = regressor.predict(X
    test_rmse = np.sqrt(mean_squared_error
    test_r2 = r2_score(y_test, test_predic

    print("Test RMSE:", test_rmse)
    print("Test NRMSE score:", test_rmse /
    print("Test R2 Score:", test_r2)
    print(f"Tuning completed in: {end_time
 ✓ 0.0s

Test RMSE: 20351.344947789778
Test NRMSE score: 0.027419891308253015
Test R2 Score: 0.9373535215331206
Tuning completed in: 105.416116 seconds.
```

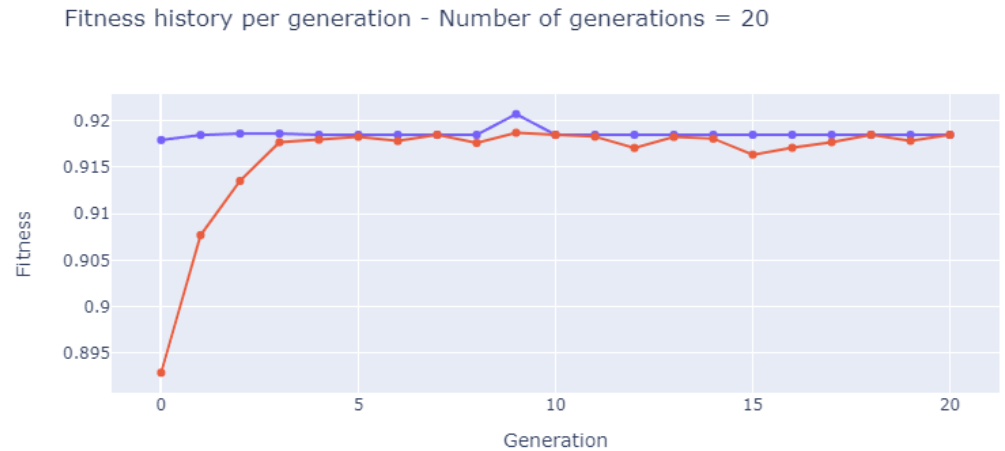
Number of generations: 30

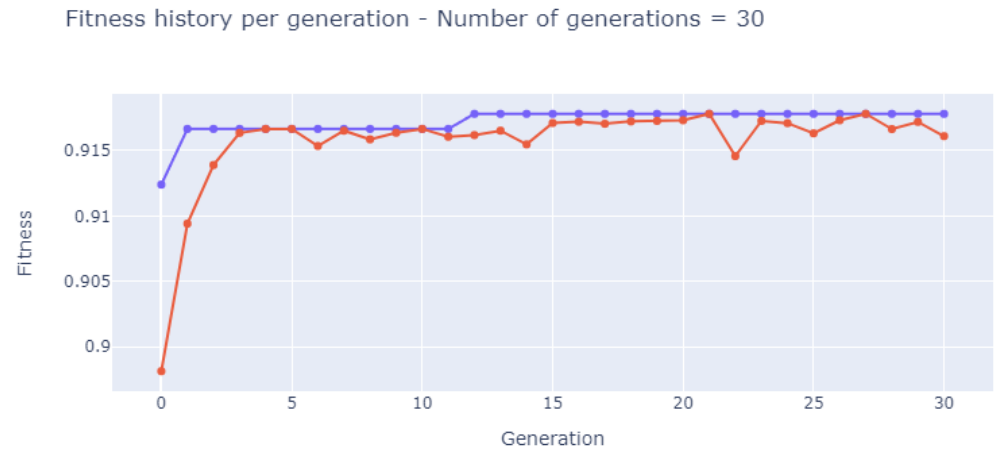
```
0.9177892784380557

    # Calculate the test metrics
    test_predictions = regressor.predict(X_
    test_rmse = np.sqrt(mean_squared_error(
    test_r2 = r2_score(y_test, test_predict

    print("Test RMSE:", test_rmse)
    print("Test NRMSE score:", test_rmse /
    print("Test R2 Score:", test_r2)
    print(f"Tuning completed in: {end_time-
 ✓ 0.0s

Test RMSE: 20832.437718805013
Test NRMSE score: 0.028068079991815015
Test R2 Score: 0.9343566681466157
Tuning completed in: 202.822427 seconds.
```

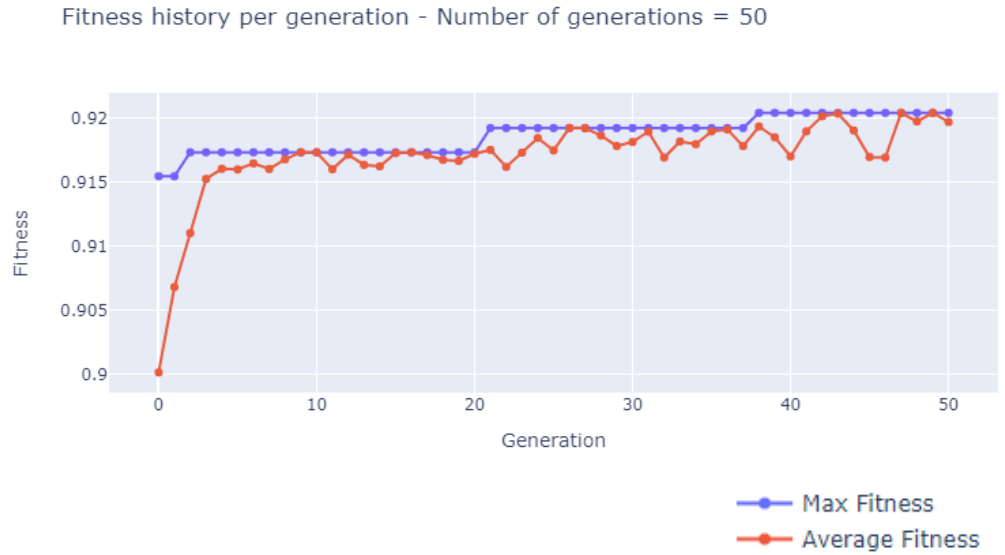
Number of generations: 50

```
0.9204213781743185

    # Calculate the test metrics
    test_predictions = regressor.predict(X_
    test_rmse = np.sqrt(mean_squared_error(
    test_r2 = r2_score(y_test, test_predict

    print("Test RMSE:", test_rmse)
    print("Test NRMSE score:", test_rmse /
    print("Test R2 Score:", test_r2)
    print(f"Tuning completed in: {end_time-
 ✓ 0.0s

Test RMSE: 20493.93461434983
Test NRMSE score: 0.027612006039185392
Test R2 Score: 0.936472593645955
Tuning completed in: 359.486092 seconds.
```


Fitness history per generation - Number of generations = 10
Fitness
Generation
0.9
0.905
0.91
0.915
0
2
4
6
8
10


Fitness history per generation - Number of generations = 20
Fitness
Generation
0.895
0.9
0.905
0.91
0.915
0.92
0
5
10
15
20


Fitness history per generation - Number of generations = 30
Fitness
Generation
0.9
0.905
0.91
0.915
0
5
10
15
20
25
30


Fitness history per generation - Number of generations = 50
Fitness
Generation
0.9
0.905
0.91
0.915
0.92
0
10
20
30
40
50
Max Fitness
Average Fitness