Predictive Model for Running Effort

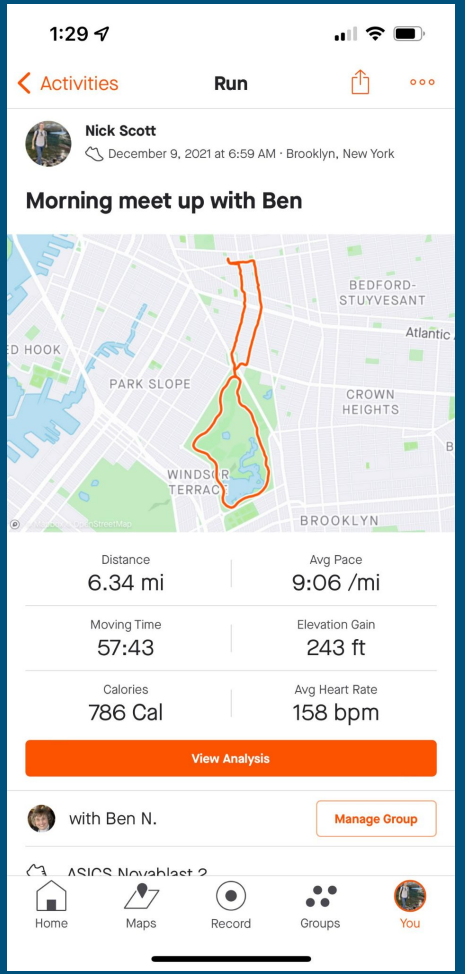
Nicholas Scott
INFO-656-01
Fall 2021

Introduction

- What is effort?
- Why are some runs harder than others?
- How could I predict how hard a given run will be before setting out?

Strava

- Popular activity and fitness tracking app for runners, cyclists, hikers, etc.
- Has an API where users can extract their data



The Idea

- Using my personal data from the past 6 months of running, I wanted to build a model that would predict the effort required to complete a run given a few inputs.
- Inputs:
  - Distance
  - Elevation
  - Pace
- Outputs
  - Average heart rate, or “effort required”

The Process

- Strava API provided perfectly “clean” data, but it required quite a bit of pre-procesing
  - All distances were given in meters
  - Instead of average “pace” (minutes per mile), the API gave “speed” in the form of “meters per second”

```
df_activities = pd.read_csv('/Users/nickx/Documents/school_work/21FA/info_656_machine_learning/ml_final_project/data/strava_activities_all_fields.csv')

df_runs = df_activities[df_activities['type'] == 'Run']
df_runs.head(10)
```

	Unnamed: 0	resource_state	name	distance	moving_time	elapsed_time	total_elevation_gain	type	workout_type	id	...	athlete.resource_state	map.id	map.summary_
0	0	2	Long run	22105.2	7208	7328	135.9	Run	0.0	6283890938	...	1	a6283890938	g~iwFhnnbM}Iz@}Ez@sCHmEz@gEHuAZr@Is@Z_D^g
1	1	2	Recovery run	6061.1	2125	2363	46.5	Run	0.0	6276813009	...	1	a6276813009	w_jwFrmnbMIBR@ @Bk@BWEAJc@TJHGDAJ@n@
2	2	2	Workout Wednesday	9850.0	3004	3018	0.0	Run	0.0	6272684257	...	1	a6272684257	kfpwFrujbMFA?IFDDCJFBI@B@GhBSJITJe@CJO_
3	3	2	Morning Run	9980.3	3159	3285	100.6	Run	0.0	6265381398	...	1	a6265381398	cjiwFtmnbMADECGBACKFSEW@gBVEFQx@UdBEFSF
4	4	2	Morning Run	9769.7	3474	3597	70.4	Run	NaN	6260936496	...	1	a6260936496	_jiwFtnnbMB@CEJO?D@IF?Ea@DBDAHJAKJEBKHD^OF
5	5	2	Easy Sunday in Philly	6602.1	2269	2280	25.9	Run	0.0	6256778448	...	1	a6256778448	ubzrFtctiMH@?HDADD@Ax@JV?V?P@VOV?b@JZTLBI
6	6	2	Friday night endurance	12758.2	4410	4749	117.3	Run	0.0	6249249996	...	1	a6249249996	a~iwF`mnbMGA_@^SJIECFCKE?EFkANQJEVE~ASdAE
7	7	2	Recovery	6155.4	2306	2445	43.6	Run	0.0	6242906333	...	1	a6242906333	cjiwF]mnbMDQCBEAEHuAZ_@Fm@PO@EFCNMTEVIdACr
8	8	2	Track workout with Bryan	10120.6	2911	3141	0.0	Run	0.0	6240896496	...	1	a6240896496	oepwFhujbMFGGTPMNFJ?NM?D`@GpA_@VMXWNYLe@
9	9	2	Tempo	9942.7	3203	3323	101.3	Run	0.0	6233556473	...	1	a6233556473	qjiwFbnnbMIFk@Hg@Ni@?i@JGDET?n@Gh@?`AEN@I

10 rows × 61 columns

The Process Continued...

- Tried using a linear, non-linear, ensemble, and NN models, but ultimately the linear model produced the most accurate result
- Training data was very limited
  - Used only my personal data, and only from a relatively recent time frame.
  - Tried to mitigate this limitation by using cross-validation

```
#trying a regression model first. Using cross-validation to try to improve accuracy since the training set is so small.
step_list_linear = [('scaler', StandardScaler()),
                    ('reg', ElasticNetCV(cv=20))
                   ]
pipe_linear = Pipeline(steps=step_list_linear)
pipe_linear
```

```
Pipeline(steps=[('scaler', StandardScaler()), ('reg', ElasticNetCV(cv=20))])
```

```
pipe_linear.fit(X_train.values, y_train.values)
```

```
Pipeline(steps=[('scaler', StandardScaler()), ('reg', ElasticNetCV(cv=20))])
```

The Final Product

- Because of low accuracy, I came up with HR “zone” function.

```
#created a function to show the heartrate "zone" for the result. This provides a little more context for the results of the model.
def effort_linear_regression(row, model):

    result = model.predict(row)

    if result <= 120:
        print("Average HR: "+str(result))
        print("Zone 1 - Easy Effort")

    if result > 120 and result <= 150:
        print("Average HR: "+str(result))
        print("Zone 2 - Endurance Effort")

    if result > 150 and result <= 170:
        print("Average HR: "+str(result))
        print("Zone 3 - Moderate Effort")

    if result > 170 and result <= 180:
        print("Average HR: "+str(result))
        print("Zone 4 - Threshold Effort")

    if result > 180:
        print("Average HR: "+str(result))
        print("Zone 5 - Max Effort")
```

```
#try the model by inputting different distances, elevations, and paces to see how "hard" it will be.
dist=4
elev=100
pace_min=9
pace_sec=20

row=np.array([dist, elev, pace_min, pace_sec])
row=row.reshape(1, -1)

effort_linear_regression(row, model)
```

```
Average HR: [164.69512201]
Zone 3 - Moderate Effort
```

Questions?

- All code and data for the project can be found on GitHub: https://github.com/nickxscott/ml_final_project
- Happy running!