GRAET

Planning Hockey Careers *with Python*

Jaroslav Bezdek · 9 December 2025

```python
AGENDA: list[str] = [...]

for i, item in enumerate(AGENDA, start=1):
    print(f"{i:02} - {item}")

# 01 - About Me
# 02 - About GRAET
# 03 - The Problem
# 04 - The Theory
# 05 - The Inference
# 06 - Summary
```

About Me

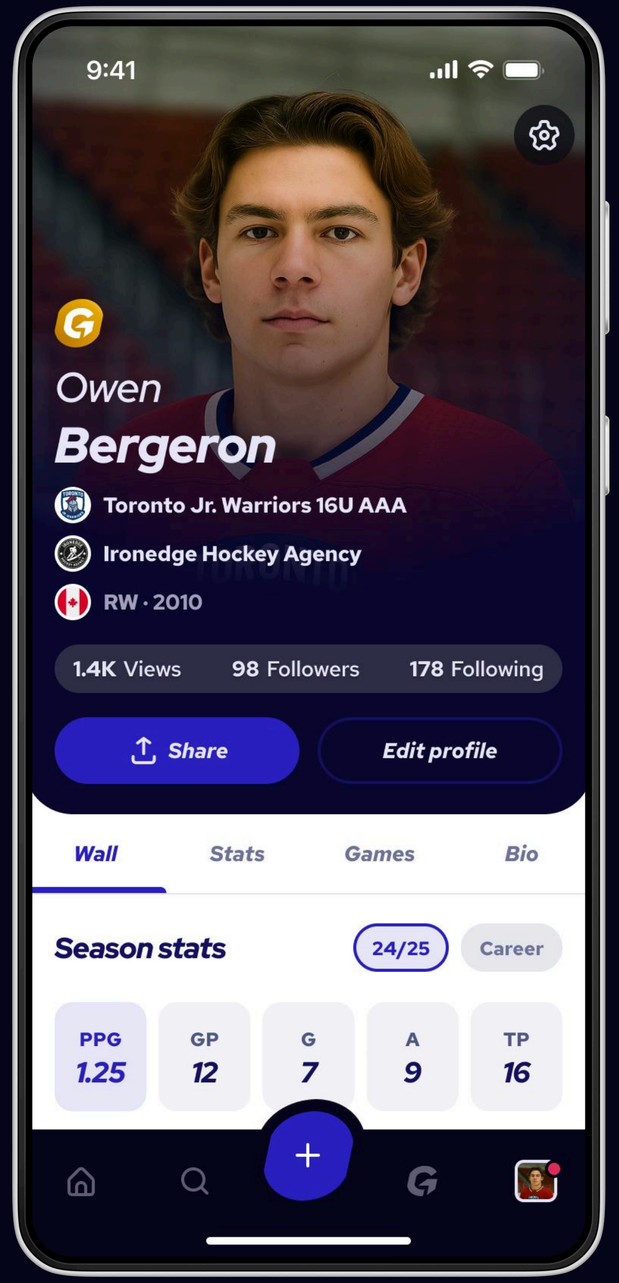
01

```
name = "Jaroslav Bezdek"
nationality = "Czech Republic"

full_time_job = "Head of Sports Intelligence"
full_time_job_company = "GRAET"

part_time_job = "Hockey Data Analyst"
part_time_job_company = "HC Sparta Prague"
```

About GRAET

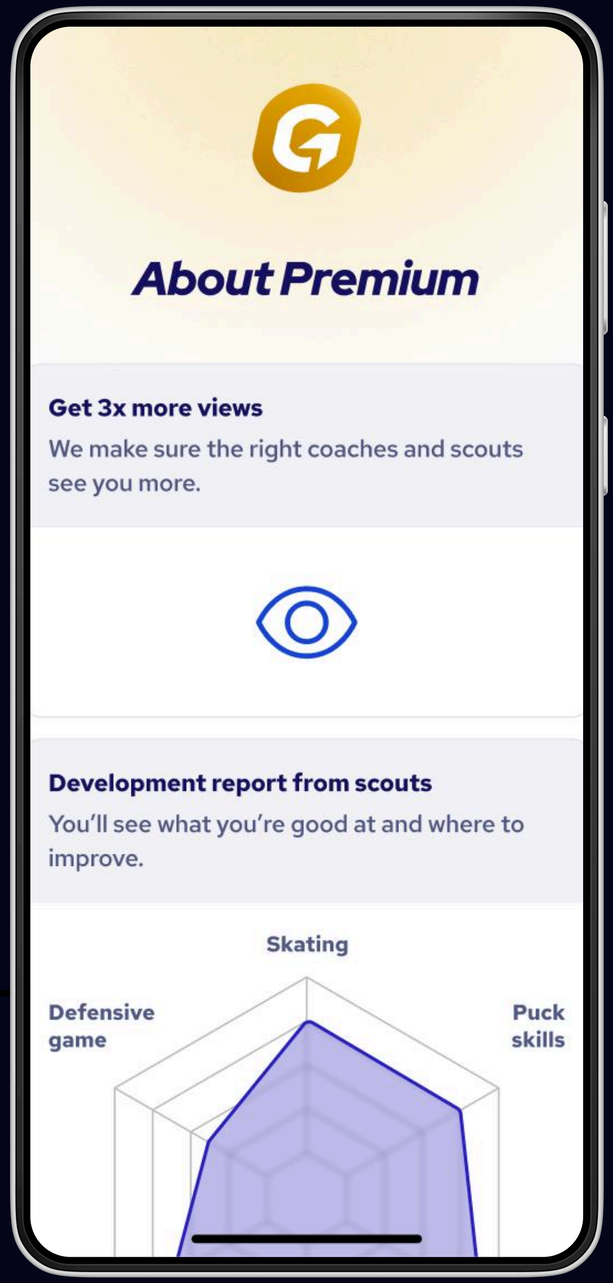
02

```
graet = MobileApp(
  nickname="LinkedIn for hockey players",
  registered_players=25_000,
)

assert graet.is_actually_great()
```



The Problem

03


About Premium
Get 3x more views
We make sure the right coaches and scouts see you more.
Development report from scouts
You'll see what you're good at and where to improve.
Skating
Defensive game
Puck skills

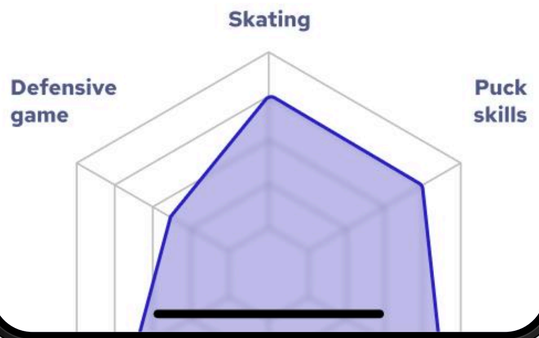
About Premium

Get 3x more views

We make sure the right coaches and scouts see you more.

Development report from scouts

You'll see what you're good at and where to improve.



9:41

Level up with a new report

Get a new development report to track your progress. We recommend ordering one every 3 months.

Order for $99

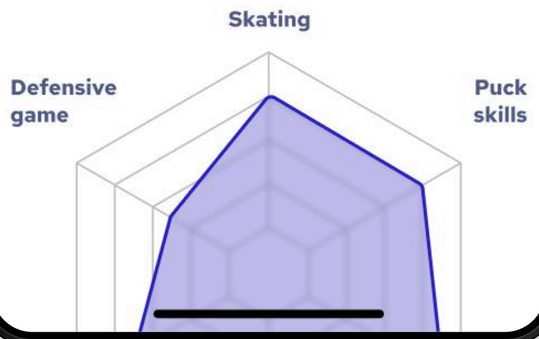
About Premium

Get 3x more views

We make sure the right coaches and scouts see you more.

Development report from scouts

You'll see what you're good at and where to improve.



9:41

Level up with a new report

Get a new development report to track your progress. We recommend ordering one every 3 months.

Order for $99

9:41

Blue Activity Career

Career path

Explore the most potentially successful paths to reach your goal and discuss each season with Blue.

Your goal
NCAA

Path #1 | Path #2 | Path #3

Season 25/26
15U AAA league

PPG target — Needs growth

0.94

0.87 1.07 1.22 1.49

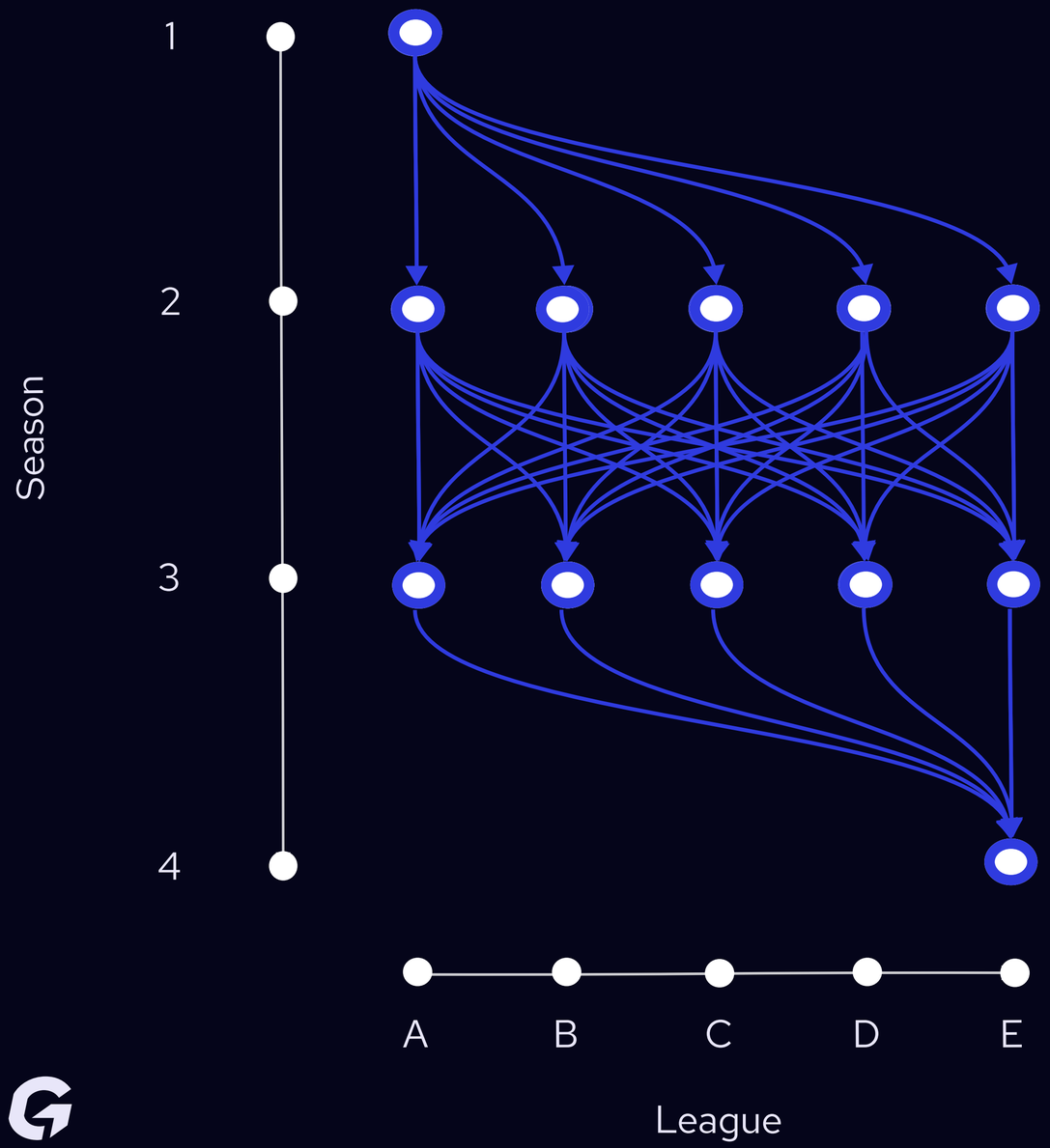
Season 26/27
16U AAA league

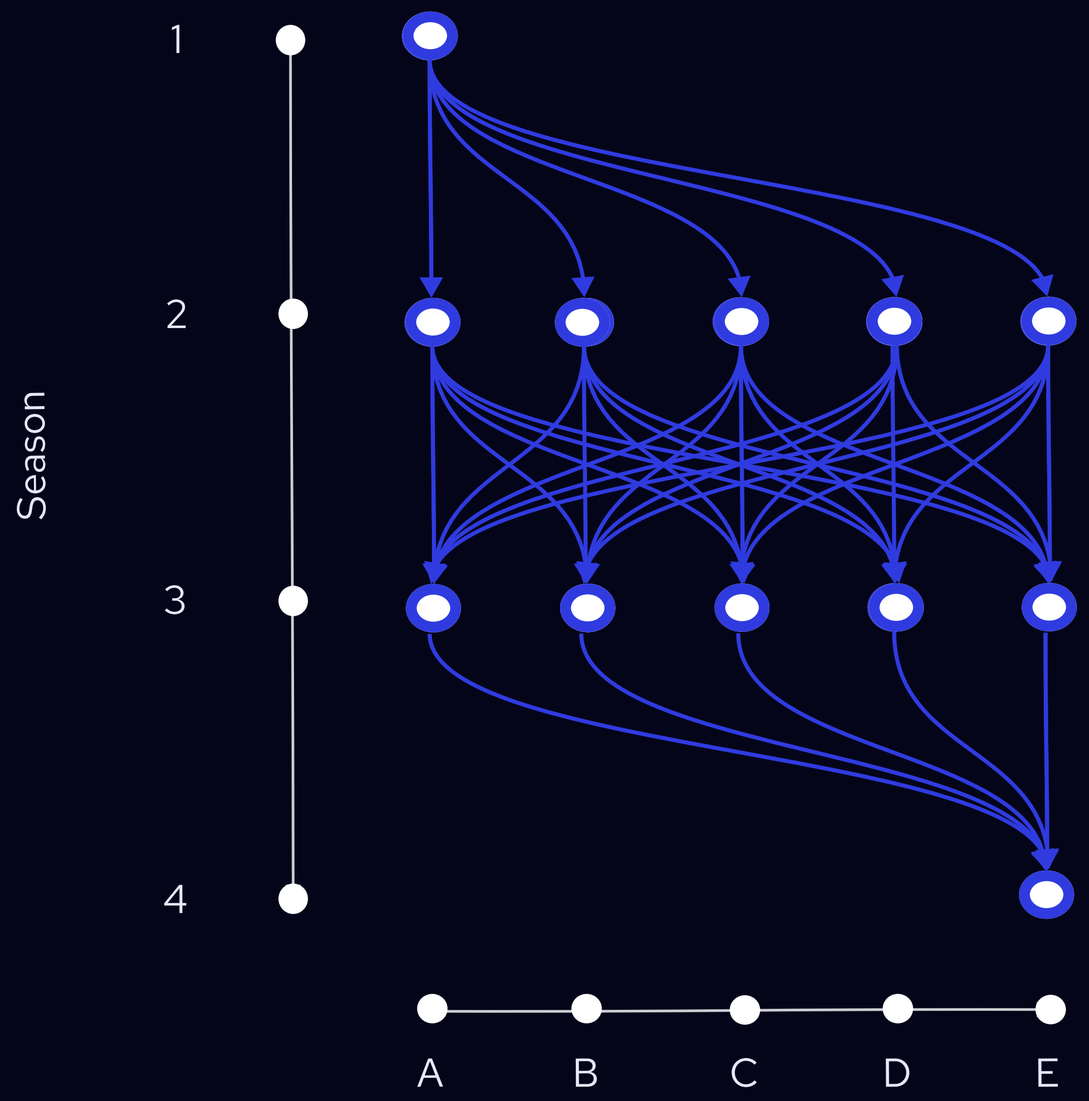
Season 27/28
NAHL

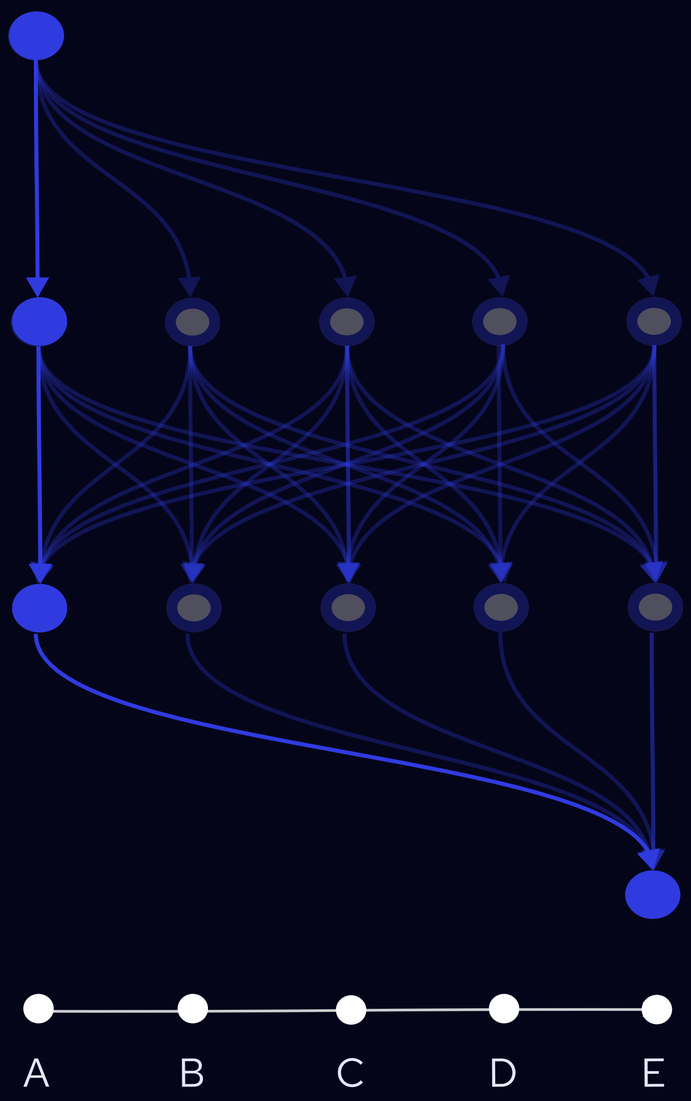
Season 28/29
NCAA

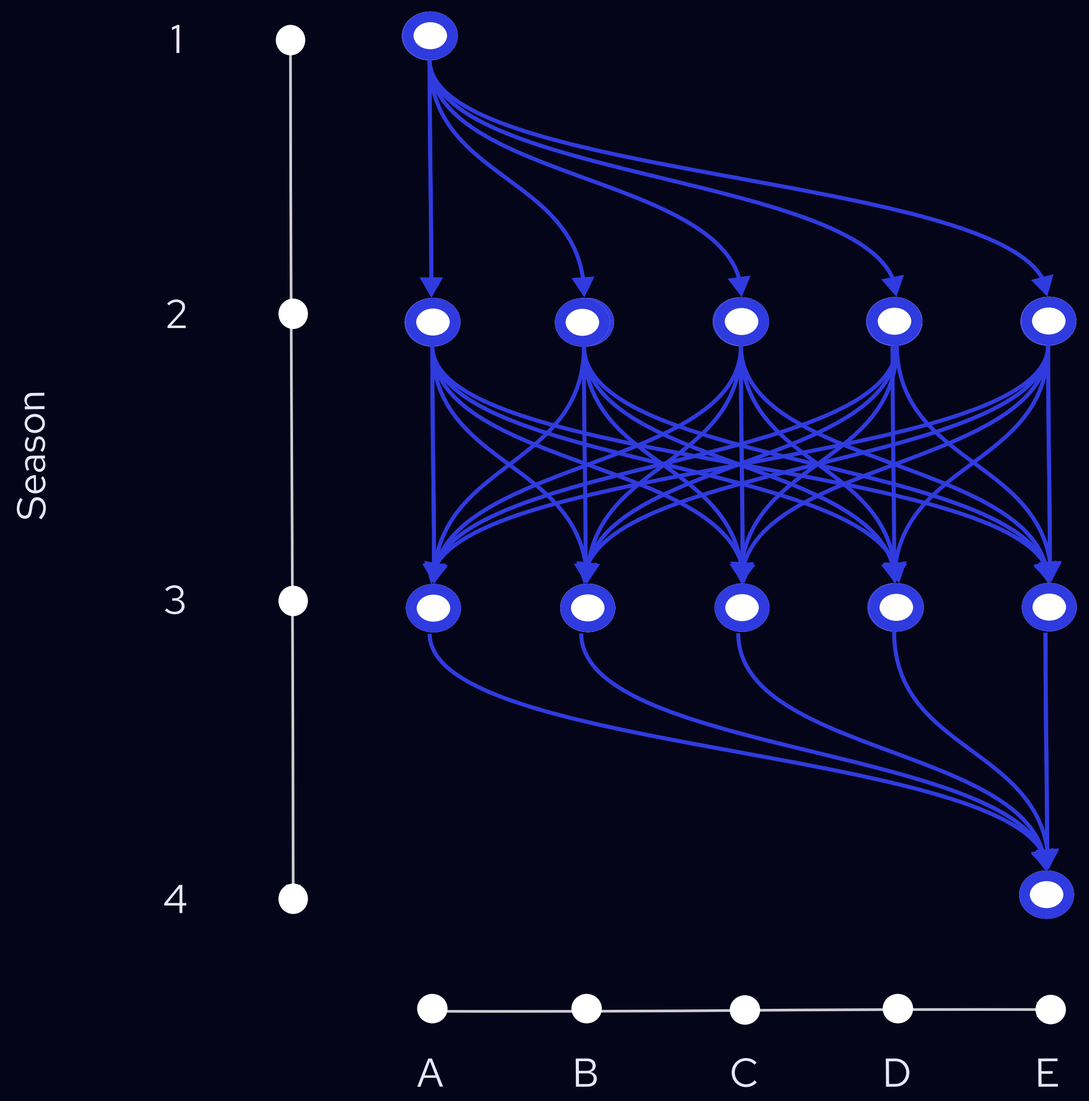
The Theory

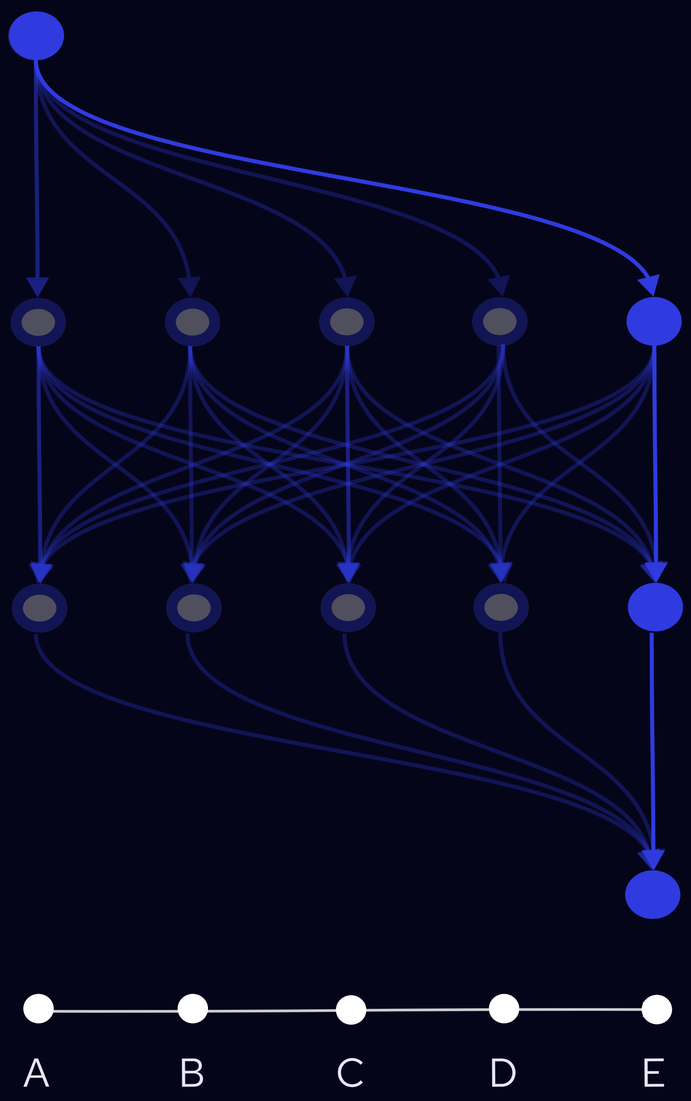
04

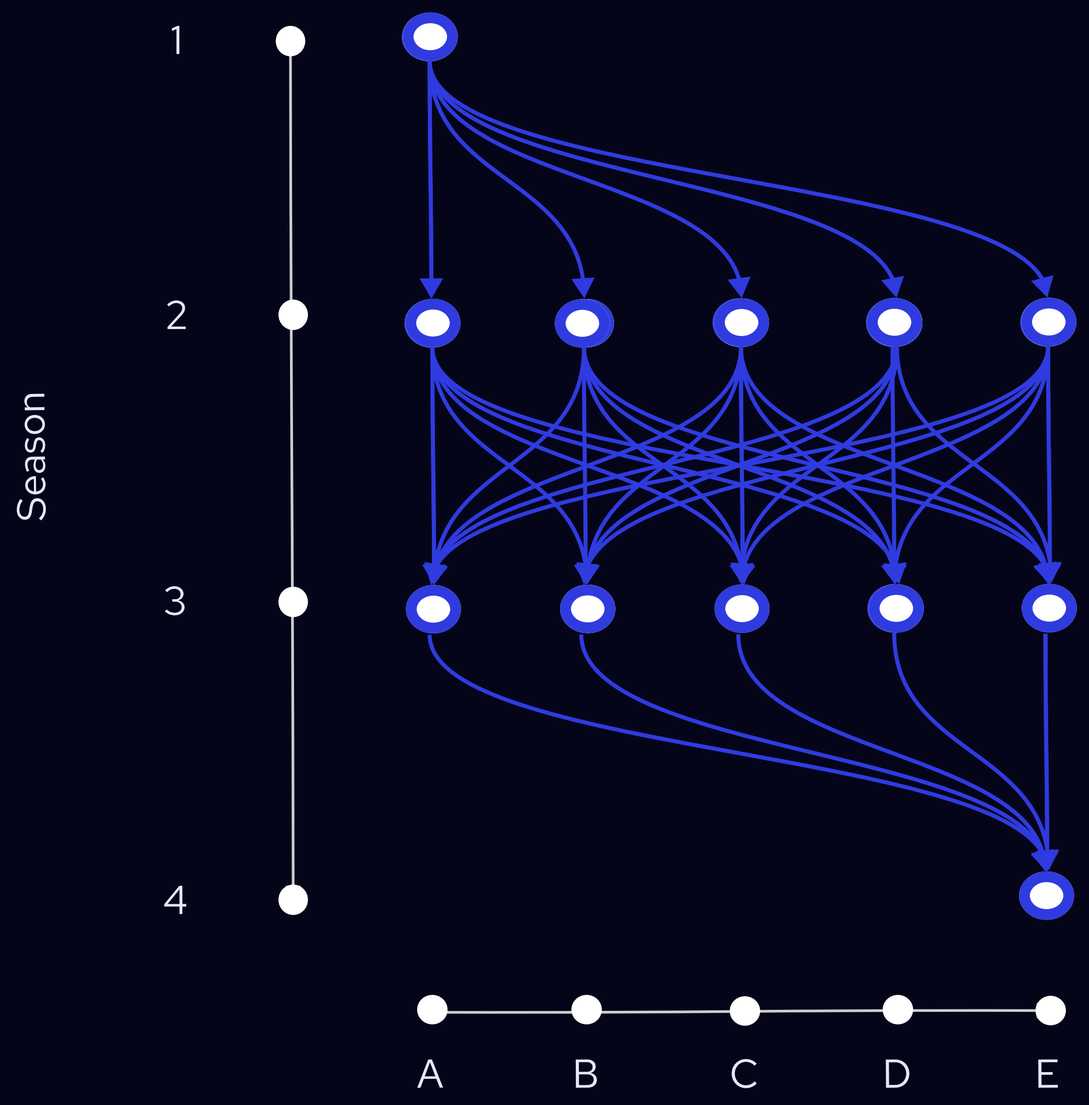

1
2
3
4
Season
A
B
C
D
E
League

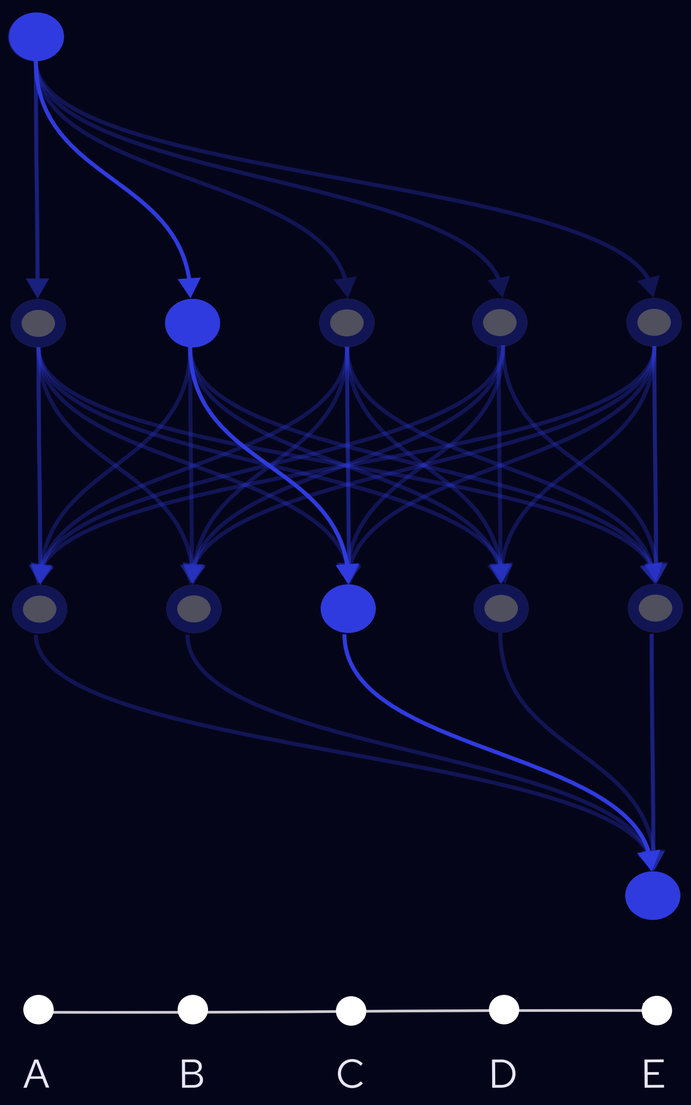

1
2
3
4
Season
A
B
C
D
E
League

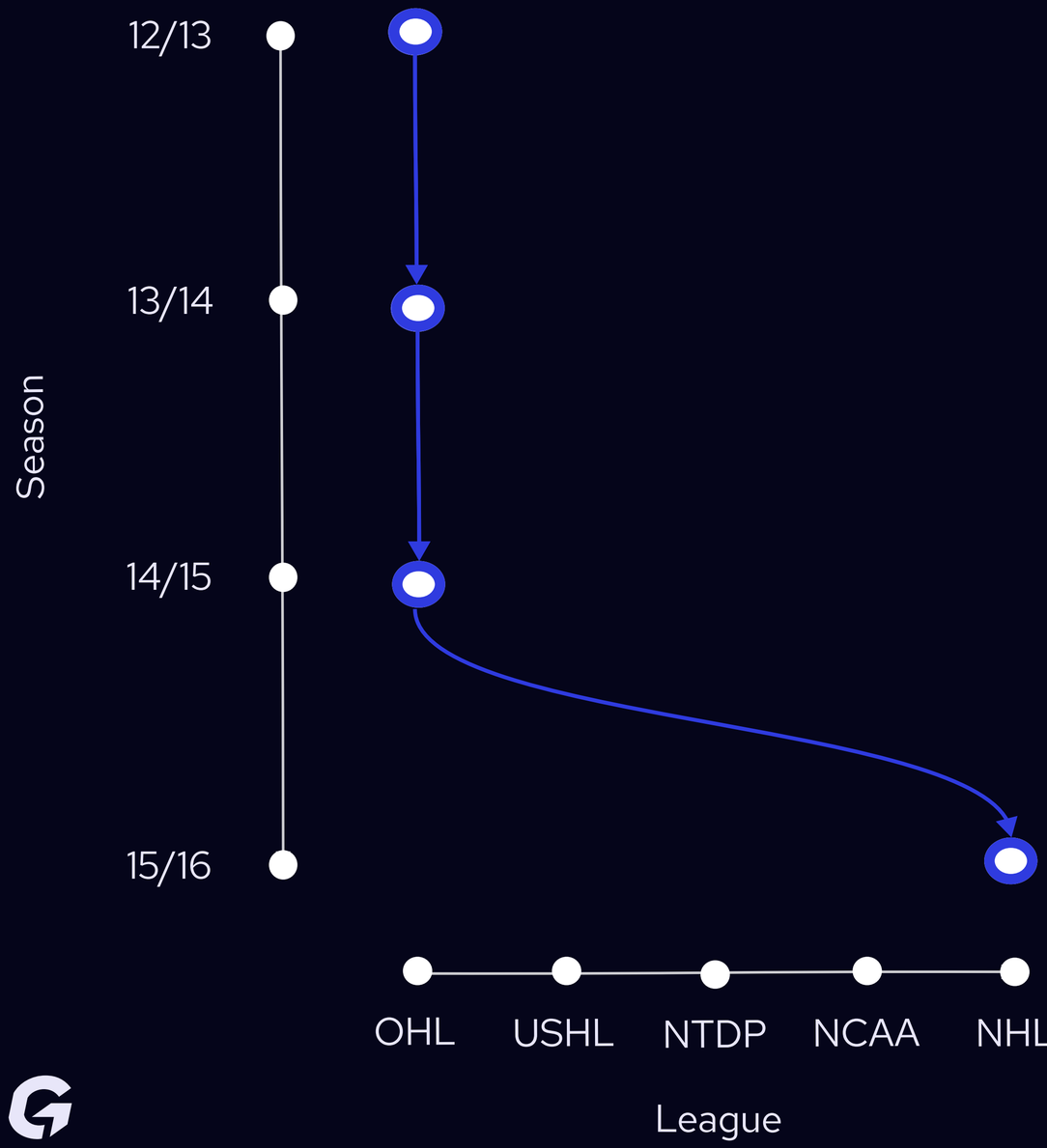

A
B
C
D
E
League


1
2
3
4
Season
A
B
C
D
E
League


A
B
C
D
E
League

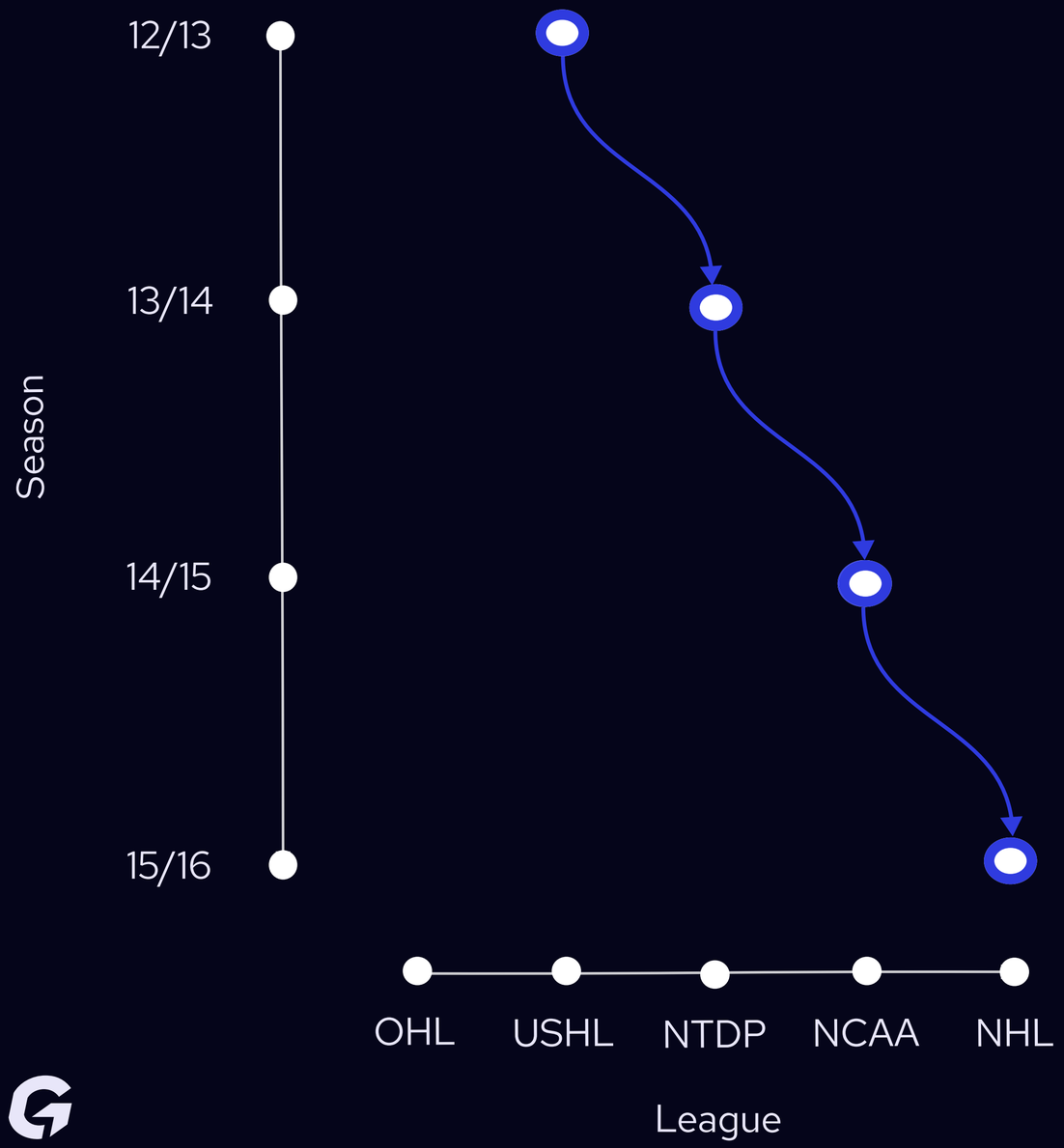

1
2
3
4
Season
A
B
C
D
E
League


A
B
C
D
E
League



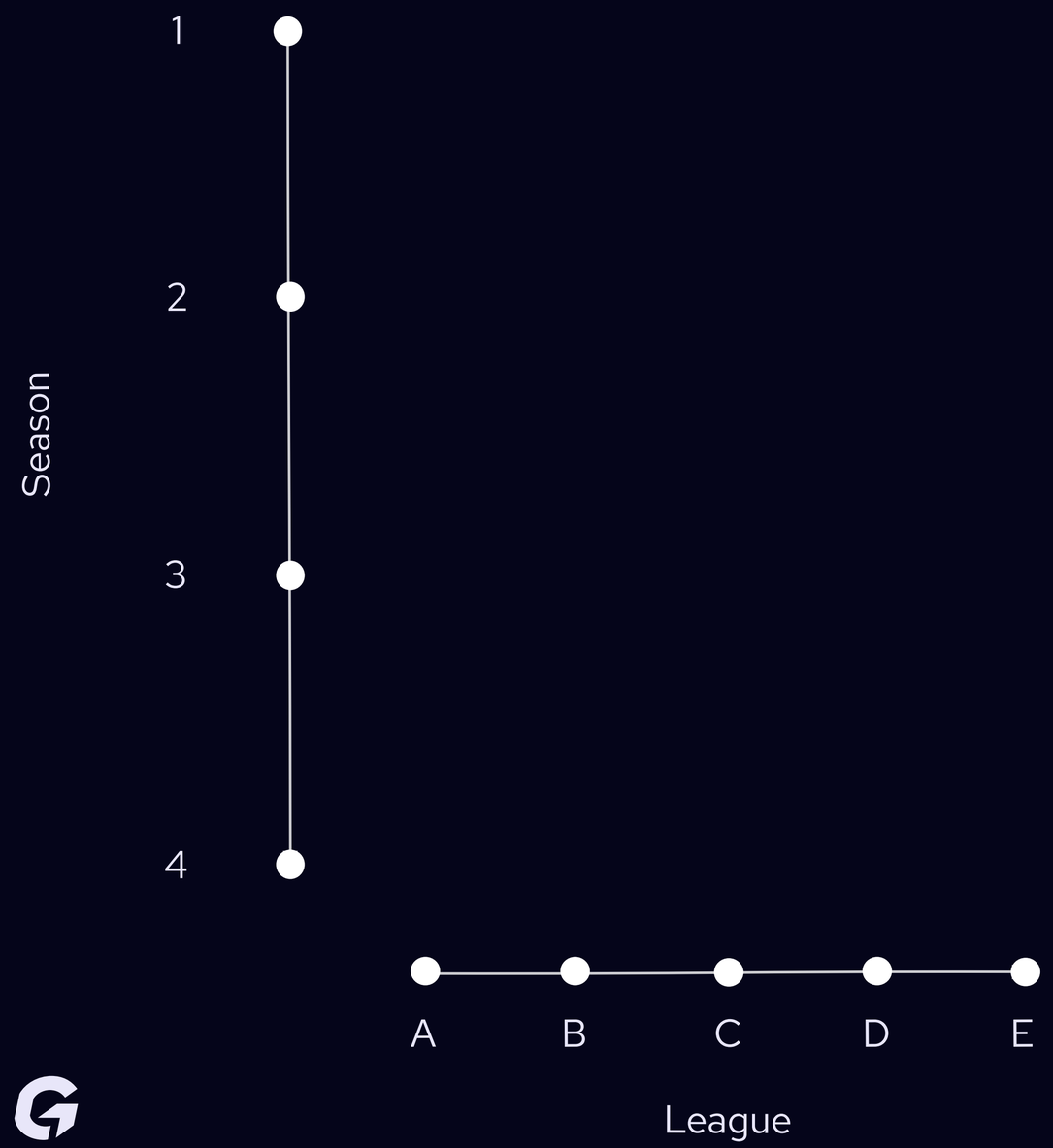
Connor McDavid

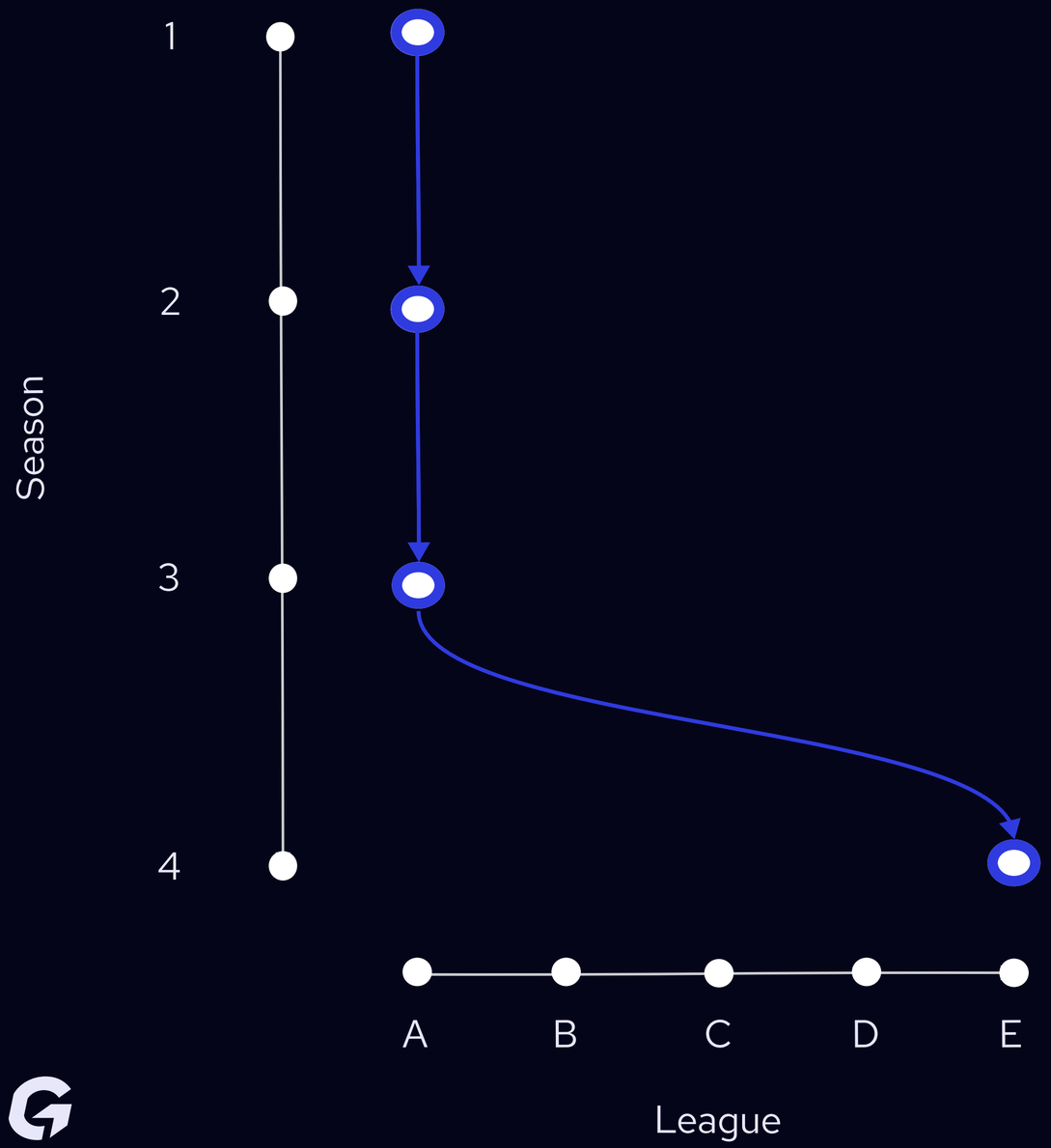
Edmonton Oilers | #97 | C

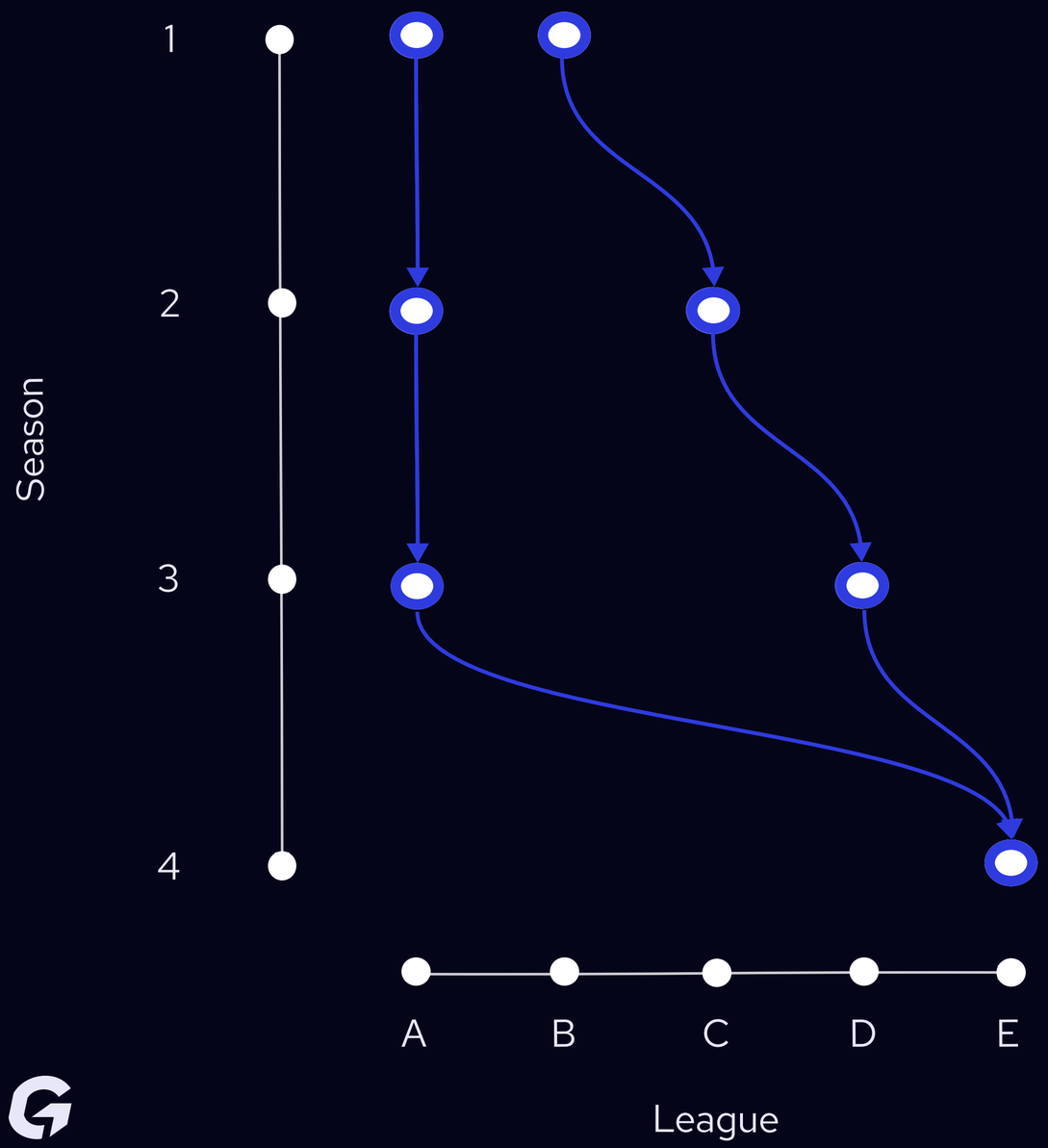


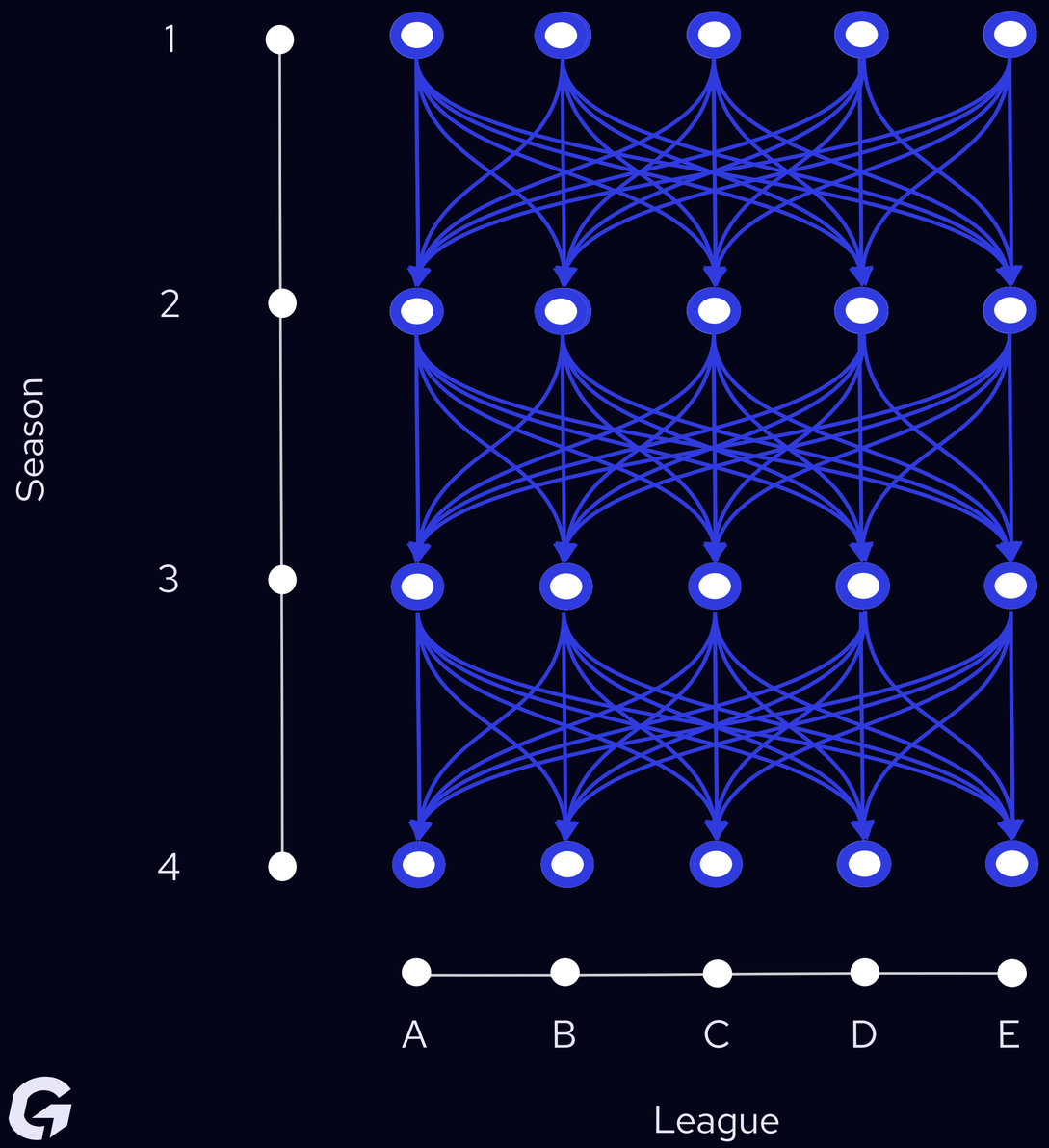
Jack Eichel

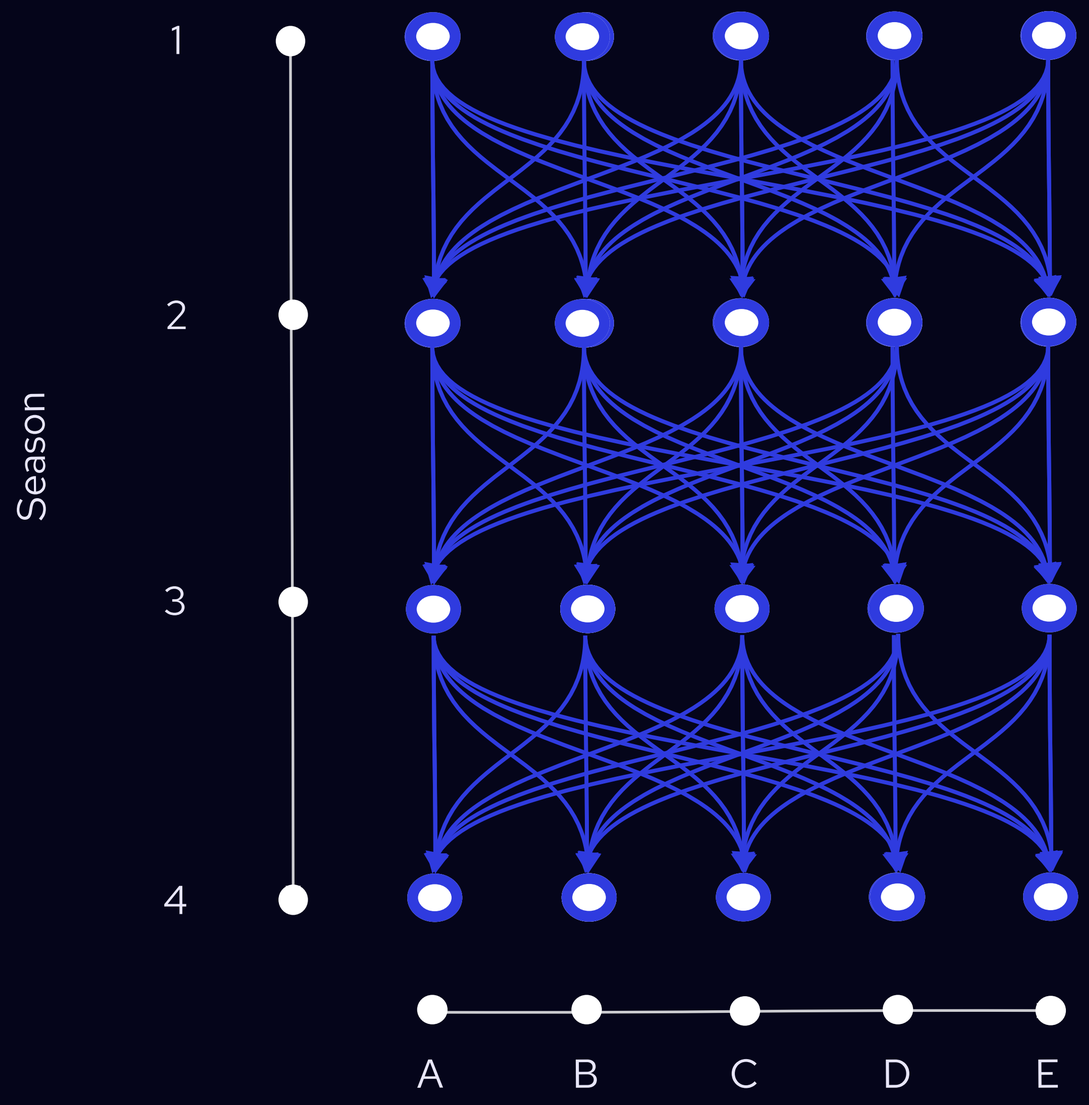
Vegas Golden Knights | #9 | C

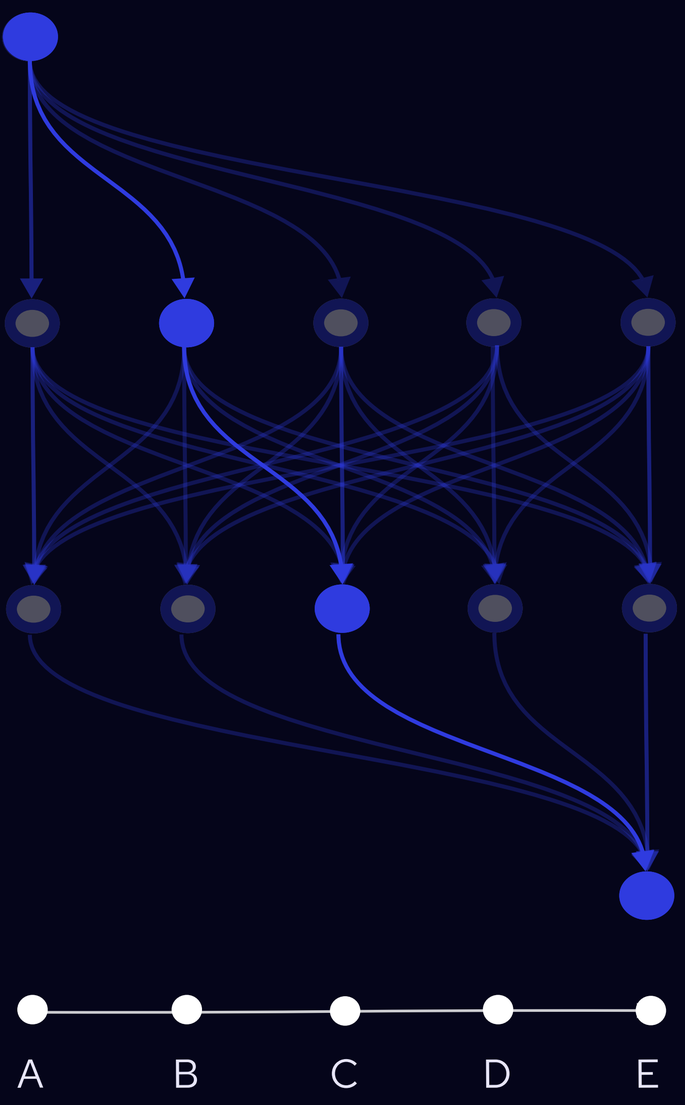

1
2
3
4
Season
A
B
C
D
E
League


1
2
3
4
Season
A
B
C
D
E
League


1
2
3
4
Season
A
B
C
D
E
League


1
2
3
4
Season
A
B
C
D
E
League


1
2
3
4
Season
A
B
C
D
E
League


A
B
C
D
E
League

The Inference

05

```python
import duckdb


df = duckdb.read_csv(path_or_buffer=...)
print(df.limit(5))
```

```
# +----------+----------+-----------+-----------------+-----+----------+
# | position | country  | handedness| age_10          | ... | age_20   |
# | ---      | ---      | ---       | ---             | ... | ---      |
# | str      | str      | str       | str             | ... | str      |
# +----------+----------+-----------+-----------------+-----+----------+
# | D        | CA       | RIGHT     | Ontario U10 AAA | ... | NHL      |
# | F        | CZ       | RIGHT     | Czechia U10     | ... | AHL      |
# | G        | CA       | RIGHT     | Quebec U11 AA   | ... | OHL      |
# | F        | SE       | RIGHT     | Sweden U10      | ... | Sweden   |
# | F        | US       | RIGHT     | Alaska U10 A    | ... | NCAA     |
# +----------+----------+-----------+-----------------+-----+----------+
```

```
request = {
    "current_age": 15,
    "current_league": "Czechia U17",
    "target_league": "NHL",
}


from operator import itemgetter


current_age, current_league, target_league = itemgetter(
    "current_age", "current_league", "target_league"
)(request)
```

```python
def get_players_with_similar_start(current_age: int, current_league: str):
    filter_expr = f"age_{current_age} = '{current_league}'"
    return df.filter(filter_expr)
```

```python
def get_players_with_similar_start(current_age: int, current_league: str):
    filter_expr = f"age_{current_age} = '{current_league}'"
    return df.filter(filter_expr)


def get_players_with_similar_target(
    current_age: int, target_league: str, max_age: int = 20
):
    filter_expr = or_conditions([
        f"age_{age} = '{target_league}'"
        for age in range(current_age + 1, max_age + 1)
    ])
    return df.filter(filter_expr)
```

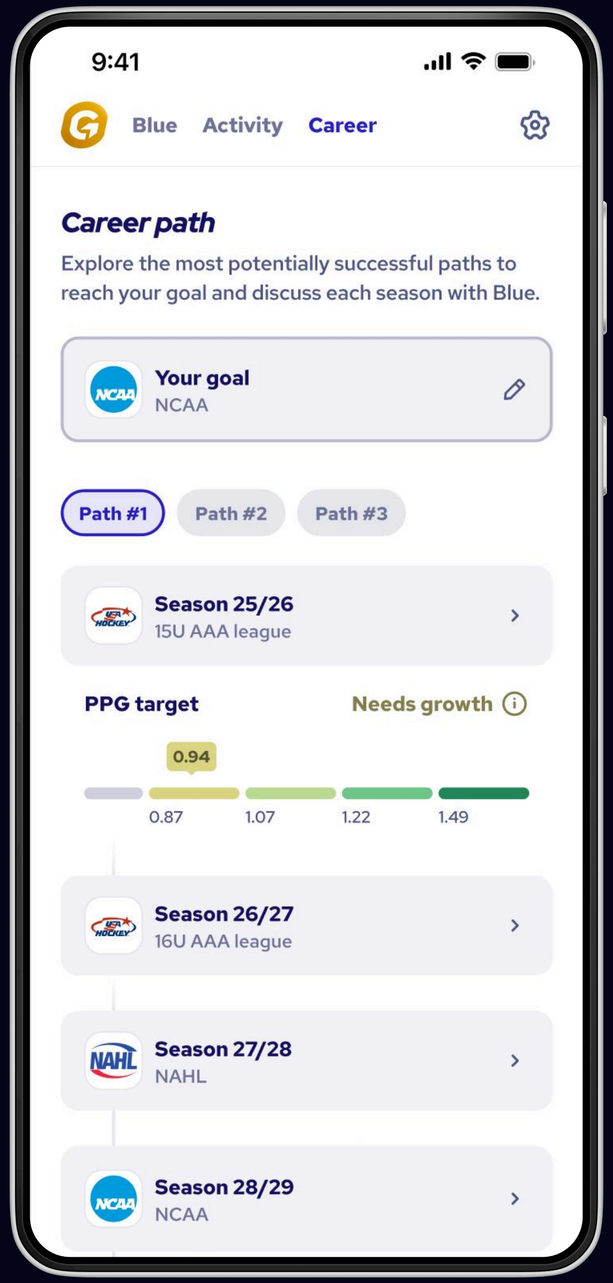
```
players_with_similar_start = get_players_with_similar_start(current_age, current_league)
players_with_similar_target = get_players_with_similar_target(current_age, target_league)


def get_similar_players(
    players_with_similar_start: duckdb.DuckDBPyRelation,
    players_with_similar_target: duckdb.DuckDBPyRelation,
):
    return duckdb.sql(
        """
        select * from players_with_similar_start
        union
        select * from players_with_similar_target
        """
    ).to_df()
```

```python
import networkx as nx


def create_graph(similar_players: pd.DataFrame):
    G = nx.DiGraph()

    age_cols = [col for col in similar_players.columns if col.startswith("age_")]

    for src_age, dst_age in zip(age_cols[:-1], age_cols[1:]):

        for row in similar_players.itertuples():
            src_node, dest_node = (src_age, row[src_age]), (dst_age, row[dst_age])
            G.add_edge(src_node, dest_node, weight=G[src_node][dest_node]["weight"] + 1)

    # compute distance
    for _, _, data in G.edges(data=True):
        data["distance"] = 1 / data["weight"]

    return G
```

```python
def find_shortest_path(
    G: nx.DiGraph, current_age: int, current_league: str, target_league: str
):
    start = (current_age, current_league)
    targets = [
        node for node in G.nodes
        if node[0] > current_age and node[1] == target_league
    ]

    # pick the cheapest path among all target nodes
    shortest_path, shortest_distance = None, float("inf")

    for target in targets:
        path = nx.shortest_path(G, start, target, weight="distance")
        distance = sum(G[u][v]["distance"] for u, v in zip(path[:-1], path[1:]))

        if distance < shortest_distance:
            shortest_path, shortest_distance = path, distance

    return shortest_path or []
```

Summary

06


9:41
Blue
Activity
Career
Career path
Explore the most potentially successful paths to reach your goal and discuss each season with Blue.
Your goal
NCAA
Path #1
Path #2
Path #3
Season 25/26
15U AAA league
PPG target
Needs growth
0.94
0.87
1.07
1.22
1.49
Season 26/27
16U AAA league
Season 27/28
NAHL
Season 28/29
NCAA

Thank you!