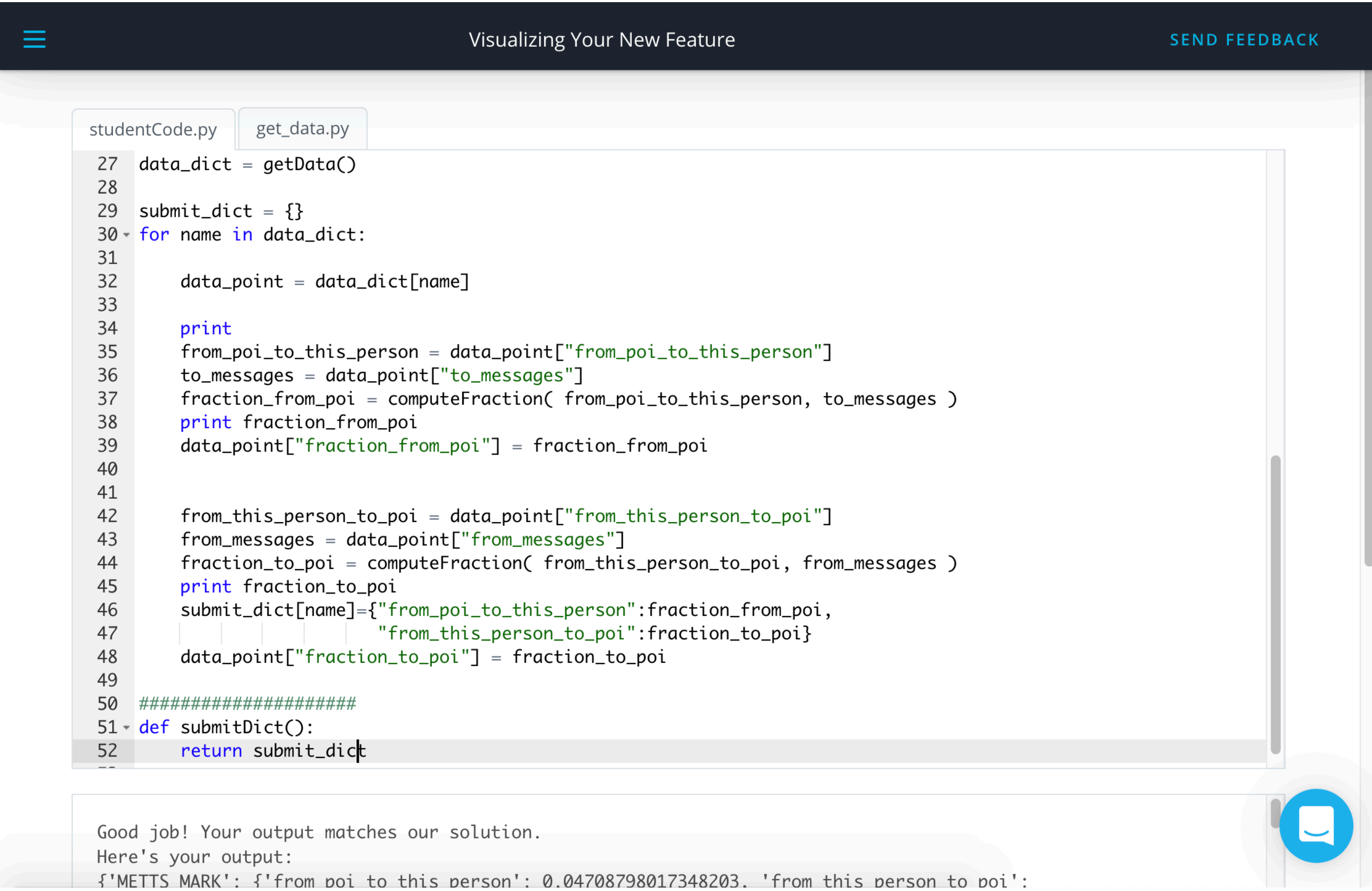
Visualizing Your New Feature

SEND FEEDBACK

studentCode.py get_data.py

```python
import pickle
from get_data import getData

def computeFraction( poi_messages, all_messages ):
    """ given a number messages to/from POI (numerator)
        and number of all messages to/from a person (denominator),
        return the fraction of messages to/from that person
        that are from/to a POI
   """


    ### you fill in this code, so that it returns either
    ###     the fraction of all messages to this person that come from POIs
    ###     or
    ###     the fraction of all messages from this person that are sent to POIs
    ### the same code can be used to compute either quantity

    ### beware of "NaN" when there is no known email address (and so
    ### no filled email features), and integer division!
    ### in case of poi_messages or all_messages having "NaN" value, return 0.
    if poi_messages == 'NaN' or all_messages == 'NaN':
        fraction = 0.
    else:
        fraction = float(poi_messages)/float(all_messages)

    return fraction
```

```
Good job! Your output matches our solution.
Here's your output:
{'METTS MARK': {'from poi to this person': 0.04708798017348203, 'from this person to poi':
```

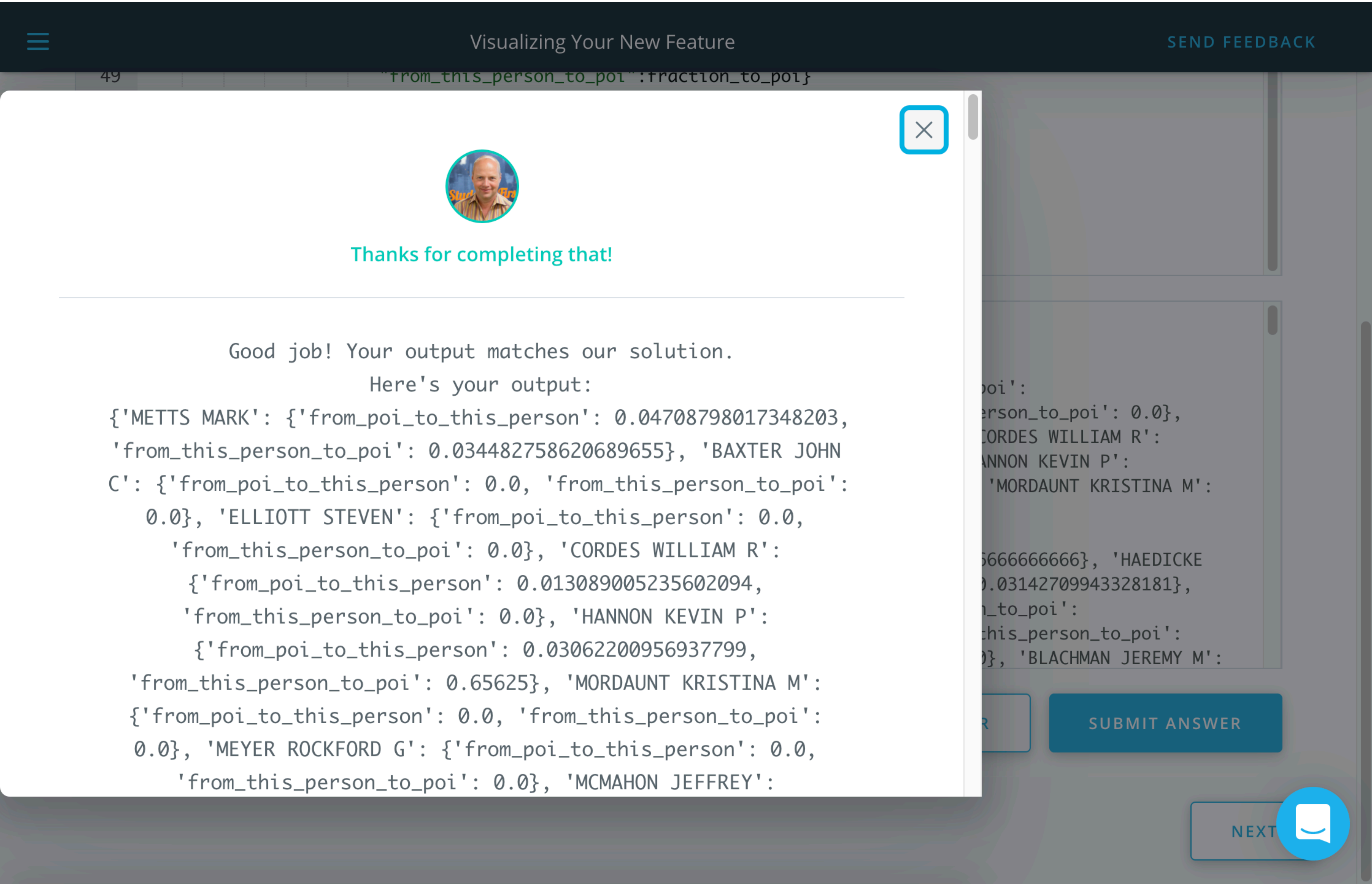

Visualizing Your New Feature

SEND FEEDBACK

studentCode.py | get_data.py

```
data_dict = getData()

submit_dict = {}
for name in data_dict:

    data_point = data_dict[name]

    print
    from_poi_to_this_person = data_point["from_poi_to_this_person"]
    to_messages = data_point["to_messages"]
    fraction_from_poi = computeFraction( from_poi_to_this_person, to_messages )
    print fraction_from_poi
    data_point["fraction_from_poi"] = fraction_from_poi


    from_this_person_to_poi = data_point["from_this_person_to_poi"]
    from_messages = data_point["from_messages"]
    fraction_to_poi = computeFraction( from_this_person_to_poi, from_messages )
    print fraction_to_poi
    submit_dict[name]={"from_poi_to_this_person":fraction_from_poi,
                       "from_this_person_to_poi":fraction_to_poi}
    data_point["fraction_to_poi"] = fraction_to_poi

#####################
def submitDict():
    return submit_dict
```

```
Good job! Your output matches our solution.
Here's your output:
{'METTS MARK': {'from_poi_to_this_person': 0.04708798017348203, 'from_this_person_to_poi':
```



```
"from_this_person_to_poi":fraction_to_poi}
```

Thanks for completing that!

```
Good job! Your output matches our solution.
Here's your output:
{'METTS MARK': {'from_poi_to_this_person': 0.04708798017348203,
'from_this_person_to_poi': 0.034482758620689655}, 'BAXTER JOHN
C': {'from_poi_to_this_person': 0.0, 'from_this_person_to_poi':
0.0}, 'ELLIOTT STEVEN': {'from_poi_to_this_person': 0.0,
'from_this_person_to_poi': 0.0}, 'CORDES WILLIAM R':
{'from_poi_to_this_person': 0.013089005235602094,
'from_this_person_to_poi': 0.0}, 'HANNON KEVIN P':
{'from_poi_to_this_person': 0.03062200956937799,
'from_this_person_to_poi': 0.65625}, 'MORDAUNT KRISTINA M':
{'from_poi_to_this_person': 0.0, 'from_this_person_to_poi':
0.0}, 'MEYER ROCKFORD G': {'from_poi_to_this_person': 0.0,
'from_this_person_to_poi': 0.0}, 'MCMAHON JEFFREY':
```

```
poi':
erson_to_poi': 0.0},
CORDES WILLIAM R':
ANNON KEVIN P':
 'MORDAUNT KRISTINA M':

5666666666}, 'HAEDICKE
).03142709943328181},
_to_poi':
this_person_to_poi':
)}, 'BLACHMAN JEREMY M':
```