Feature Request 1

#597 [Performance] Fibonacci Heaps for Ball Tree

Link: https://github.com/scikit-learn/scikit-learn/issues/597

Description:

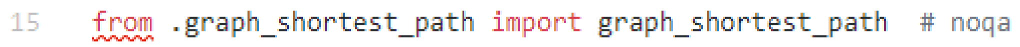
The Ball Tree algorithm is a method of performing the Nearest Neighbor search. Current implementation relies on Dijkstra’s algorithm which achieves its peak theoretical speed with a Fibonacci heap. So Ball Tree Nearest Neighbor search could potentially be sped up using Fibonacci heaps instead of the current implementation.

- an implementation of Fibonacci heap already exists in the sklearn/utils/graph_nearest_neighbor.pyx cython file.
- It would be refactored to fit requirements, and be called whenever Dijkstra’s algorithm is used. This solution would “catch” all instances of different classification algorithms involving BallTree.
- Dijkstra’s algorithm is only used in the graph_shortest_path.pyx file.

Changes:

BallTree itself requires no change, it is only when performing the Nearest Neighbor search on a constructed BallTree instance, that we want to change the data structures used.

In the graph.py utility file, graph_shortest_path is imported:

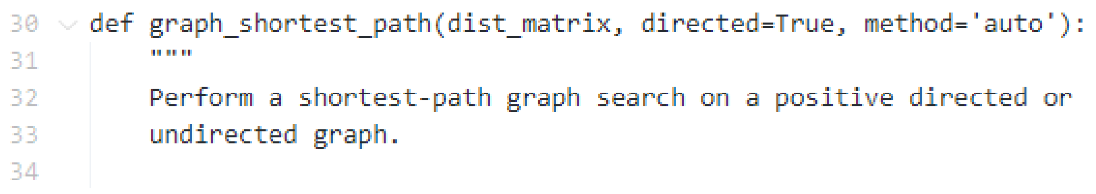


```
from .graph_shortest_path import graph_shortest_path  # noqa
```

The implementation of Dijkstra’s algorithm regarding graph_shortest_path is found in the graph_shortest_path.pyx

File: https://github.com/scikit-learn/scikit-learn/blob/main/sklearn/utils/graph_shortest_path.pyx

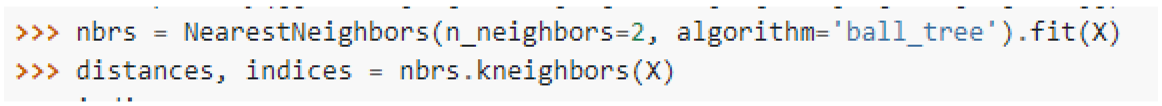
In this file we want to change the current implementation of min_heap into fib_heap.



```
def graph_shortest_path(dist_matrix, directed=True, method='auto'):
    """
    Perform a shortest-path graph search on a positive directed or
    undirected graph.

```

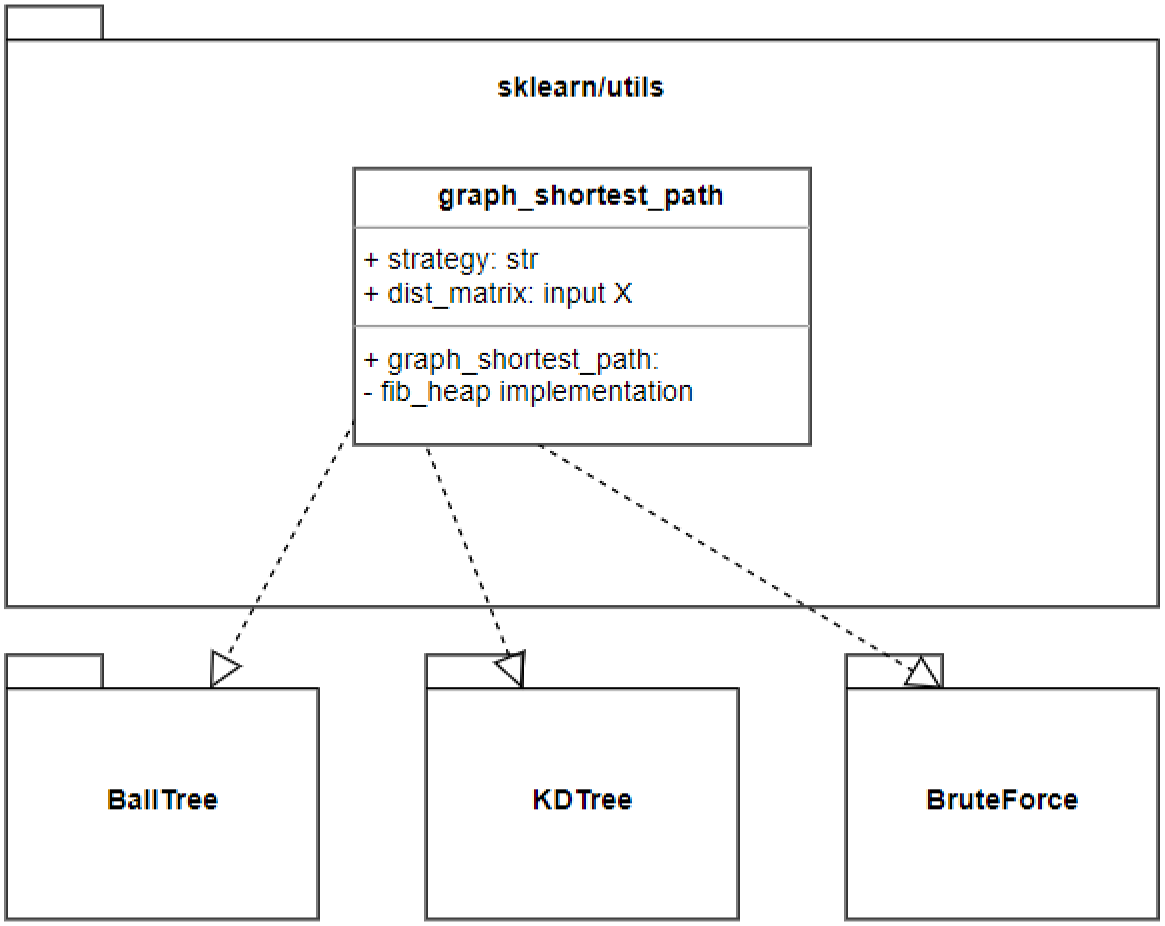
When successfully implemented, the ammortized runtime of Nearest Neighbor should be lowered.



```
>>> nbrs = NearestNeighbors(n_neighbors=2, algorithm='ball_tree').fit(X)
>>> distances, indices = nbrs.kneighbors(X)
```

Diagram:

All implementation methods are within the class of graph_shortest_path


sklearn/utils
graph_shortest_path
+ strategy: str
+ dist_matrix: input X
+ graph_shortest_path:
- fib_heap implementation
BallTree
KDTree
BruteForce