# GetIntersection

## Request and response syntax

A `GetIntersection` call takes as input a `GfsIntersectionRequest` thrift struct.

```thrift

struct GfsIntersectionRequest {

1: required i64 userId

2: required list<i64> candidateUserIds

3: required list<FeatureType> featureTypes

}

```

The response is returned in a `GfsIntersectionResponse` thrift struct.

```thrift

struct GfsIntersectionResponse {

1: required i64 userId

2: required list<GfsIntersectionResult> results

}

struct GfsIntersectionResult {

1: required i64 candidateUserId

2: required list<IntersectionValue> intersectionValues

}

struct IntersectionValue {

1: required FeatureType featureType

2: optional i32 count

3: optional list<i64> intersectionIds

4: optional i32 leftNodeDegree

5: optional i32 rightNodeDegree

}(persisted="true")

```

## Behavior

The `GfsIntersectionResponse` contains in its `results` field a `GfsIntersectionResult` for every candidate in `candidateIds` which contains an `IntersectionValue` for every `FeatureType` in the request's `featureTypes` field.

The `IntersectionValue` contains the size of the intersection between the `leftEdgeType` edges from `userId` and the `rightEdgeType` edges from `candidateId` in the `count` field, as well as their respective degrees in the graphs in `leftNodeDegree` and `rightNodeDegree` respectively.

\*\*Note:\*\* the `intersectionIds` field currently only contains `Nil`.