Tell us what your idea is.

Describe in 250 words what the feature or service will do and how you'll use Machine Learning to push the bar:

Have you ever been on a road trip and had no idea where to stop or what to see? Fear not, Sightsee is here! I'm going to refresh an existing road trip planner I created! As it exists now, users can select the types of waypoints they want to see (based on "types" in the Google Places API) and the distance in miles they're willing to travel from their route. Sightsee then generates waypoints that meet the selected criteria within the specified distance. Users can add or remove these waypoints from their route. Sightsee then redraws the user's route and provides an option to direct the user to Google Maps for directions.

How am I going to use Machine Learning? Instead of using the manual waypoint selection, I'll train a model to predict a user's possible places of interest. That way, the app can look at how far you're willing to travel and automatically generate waypoints based on your interests. I'm still researching the best places to get my training data, but I want to find something like Yelp's dataset. Right now, this app stands by itself. But wouldn't it be cool to have this capability in Google Maps? I would love to have a "Road Trip" feature that allows me to add stops and suggests new ones based on my Google reviews.

Tell us how you plan on bringing it to life.

Describe where your project is, how you could use Google's help in the endeavor, and how you plan on using On-Device ML technology to bring the concept to life. The best submissions have a great idea combined with a concrete path of where you plan on going, which should include:

- *(1) any potential sample code you've already written,*
- *(2) a list of the ways you could use Google's help,*
- *(3) as well as the timeline on how you plan on bringing it to life by May 1, 2020.*

The first stage of my project is functional! Users can choose two "adventures", so to speak. They can select an origin and destination and generate waypoints along the route, but they can also find points of interest within a specified distance of their current location. Logged in users also have the ability to save, share, and delete trips.

Here's the Douglas-Peucker algorithm at work in my app! This algorithm is really at the heart of my project. It works by looking at a Polyline (the route) and creating a similar line with fewer points. This is helpful because it reduces the number of API calls I end up making.

SNIPPETS OF WHAT'S GOING ON RIGHT NOW:

*** credit to github user aryoxp for a lot of the boilerplate logic!**

Creating boxes that are bounded by the user-specified distance. Finding the center of that box and passing it to the loadNearbyPlaces function to generate waypoints within the specified distance.

```
public void onRouteBoxerGrid(ArrayList<RouteBoxer.Box> boxes, int boxBorderColor, int markedColor, int
simpleMarkedColor) {
  if (this.gridBoxes == null)
     this.gridBoxes = new ArrayList<>();
  else this.gridBoxes.clear();

  centerBoxes = new ArrayList<LatLng>();
  for (RouteBoxer.Box box : boxes) {
     LatLng nw = new LatLng(box.ne.latitude, box.sw.longitude);
     LatLng se = new LatLng(box.sw.latitude, box.ne.longitude);
     LatLng sw = new LatLng(box.sw.latitude, box.sw.longitude);
     LatLng ne = new LatLng(box.ne.latitude, box.ne.longitude);
       if (box.marked) {
        LatLng center = new LatLng((box.ne.latitude + box.sw.latitude)/2, (box.ne.longitude +
box.sw.longitude)/2);
        loadNearByPlaces(center.latitude, center.longitude);
     }
  }
}
```

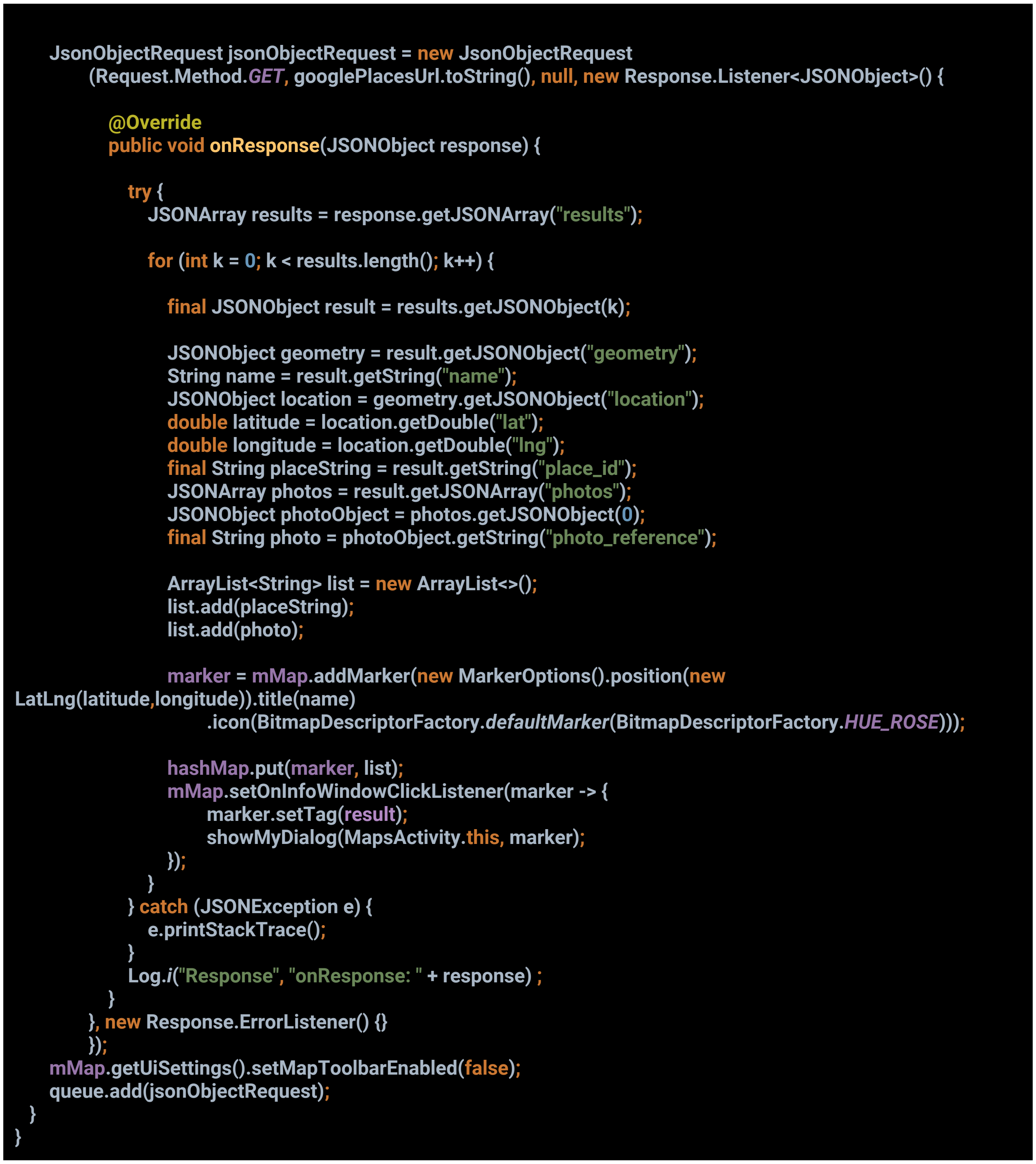
loadNearbyPlaces function that makes an API call to find selected Place types within a certain distance

```
private void loadNearByPlaces(double latitude, double longitude) {
  ArrayList<String> listPlaces = MyAdapter.getSelectedString();

  for (int i = 0; i < listPlaces.size(); i++) {
     String type = listPlaces.get(i);
     StringBuilder googlePlacesUrl =
           new StringBuilder("https://maps.googleapis.com/maps/api/place/nearbysearch/json?");
     googlePlacesUrl.append("location=").append(latitude).append(",").append(longitude);
     googlePlacesUrl.append("&radius=").append(distance);
     googlePlacesUrl.append("&type=").append(type);
     googlePlacesUrl.append("&key=" + getString(R.string.google_maps_key));

     Log.d("loadnearby", "loadNearByPlaces: " + googlePlacesUrl);

     RequestQueue queue = Volley.newRequestQueue(this);
```



```
    JsonObjectRequest jsonObjectRequest = new JsonObjectRequest
            (Request.Method.GET, googlePlacesUrl.toString(), null, new Response.Listener<JSONObject>() {

                @Override
                public void onResponse(JSONObject response) {

                    try {
                        JSONArray results = response.getJSONArray("results");

                        for (int k = 0; k < results.length(); k++) {

                            final JSONObject result = results.getJSONObject(k);

                            JSONObject geometry = result.getJSONObject("geometry");
                            String name = result.getString("name");
                            JSONObject location = geometry.getJSONObject("location");
                            double latitude = location.getDouble("lat");
                            double longitude = location.getDouble("lng");
                            final String placeString = result.getString("place_id");
                            JSONArray photos = result.getJSONArray("photos");
                            JSONObject photoObject = photos.getJSONObject(0);
                            final String photo = photoObject.getString("photo_reference");

                            ArrayList<String> list = new ArrayList<>();
                            list.add(placeString);
                            list.add(photo);

                            marker = mMap.addMarker(new MarkerOptions().position(new
LatLng(latitude,longitude)).title(name)
                                    .icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_ROSE)));

                            hashMap.put(marker, list);
                            mMap.setOnInfoWindowClickListener(marker -> {
                                    marker.setTag(result);
                                    showMyDialog(MapsActivity.this, marker);
                            });
                        }
                    } catch (JSONException e) {
                        e.printStackTrace();
                    }
                    Log.i("Response", "onResponse: " + response) ;
                }
            }, new Response.ErrorListener() {}
            });
    mMap.getUiSettings().setMapToolbarEnabled(false);
    queue.add(jsonObjectRequest);
  }
}
```

Here's Where I Need Your Help!

- I've been taking a machine learning course in my free time, but I don't know what outcomes are realistic. What data can I have access to and use? What makes good training data?
- How can I use TensorFlow Lite to create the type of model I want to use?
- How can use the demographics of my user to generate personalized suggestions?
- How can I best optimize my use of Google APIs?
- How can I integrate other APIs with what I'm already doing? For example, I'd love to use Geocaching's API to add cache locations to the map.

Timeline:

Pre-work - fix existing bugs in waypoint dialogue boxes, write more tests, rework front end as time allows

- **Sprint 1 (February 1 - 15)**
  - Research/find training data
  - Rework front end to create a more modern display
- **Sprint 2 (February 15 - 29)**
  - Train my model(s)
    - If complete, evaluate results and possibly retrain
  - Continue styling work
- **Sprint 3 (February 29 - March 14)**
  - Deploy my model(s) and see what happens
    - If disaster, step back and reevaluate
  - Continue styling work
  - Write more tests
- **Sprint 4 (March 14 - March 28)**
  - **Integration!** Marry my existing app and my trained model(s)
  - Write tests
- **Sprint 5 (March 28 - April 11)**
  - Continue integration
  - Continue tests
  - Begin user tests
- **Sprint 6 (April 11 - April 25)**
  - Continue user tests
  - Do final pass of style and tests
- **Sprint 7 (April 25 - May 1)**
  - Do final pass of user experience/make changes as needed
  - Publish to Google Play!

Tell us about you.

A great idea is just one part of the equation; we also want to learn a bit more about you. Share with us some of your other projects so we can get an idea of how we can assist you with your project.

I'm a career changer! I studied Bioethics in undergrad, and worked at nonprofits and in the legal field for a few years after college. I kept running up against problems that organizations didn't have the bandwidth or desire to solve. After years of this, I thought, "If I can spot the problems, why can't I fix them?" So I applied to Ada Developers Academy in Seattle, and the rest is history (in the making).

One of my favorite projects is the one that I want to revamp for this challenge - Sightsee was the first app I ever built, and I only had a month to build it. It was such an enjoyable experience and really cemented my love of development.

Ethics are important to me (I basically majored in it!), and machine learning is the perfect storm of ethics and technology. The ability to make things "for good" is a skill that I want to take with me throughout my career. After my internship at Nordstrom, I'm joining a machine learning team!

A cool thing that I built at Nordstrom is an auditing application that recursively traverses an n-ary tree and sends logs and metrics if the values of the child nodes do not add up to the parent nodes. I loved this because I actually got to use my computer science fundamentals in a real-world project! I'm really proud of this project because it's simple and beautiful.

Next steps.

- Be sure to include this cover letter in your GitHub repository
- Your GitHub repository should be tagged #AndroidDevChallenge
- Don't forget to include other items in your GitHub repository to help us evaluate your submission; you can include prior projects you've worked on, sample code you've already built for this project, or anything else you think could be helpful in evaluating your concept and your ability to build it
- **The final step is to fill out this form to officially submit your proposal.**

ANDROID
DEVELOPER
CHALLENGE