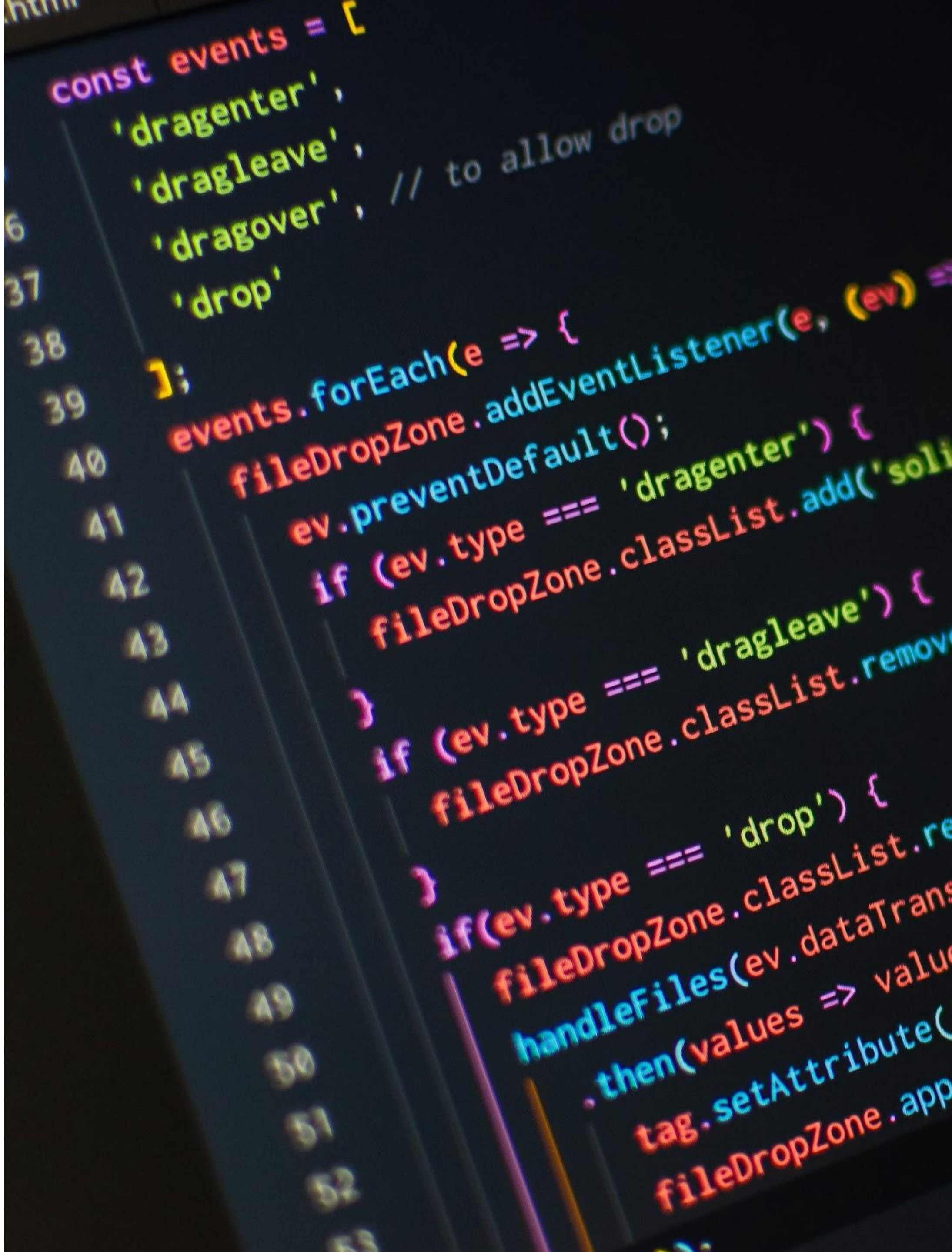
Interpreting Code

with Natural Language Processing

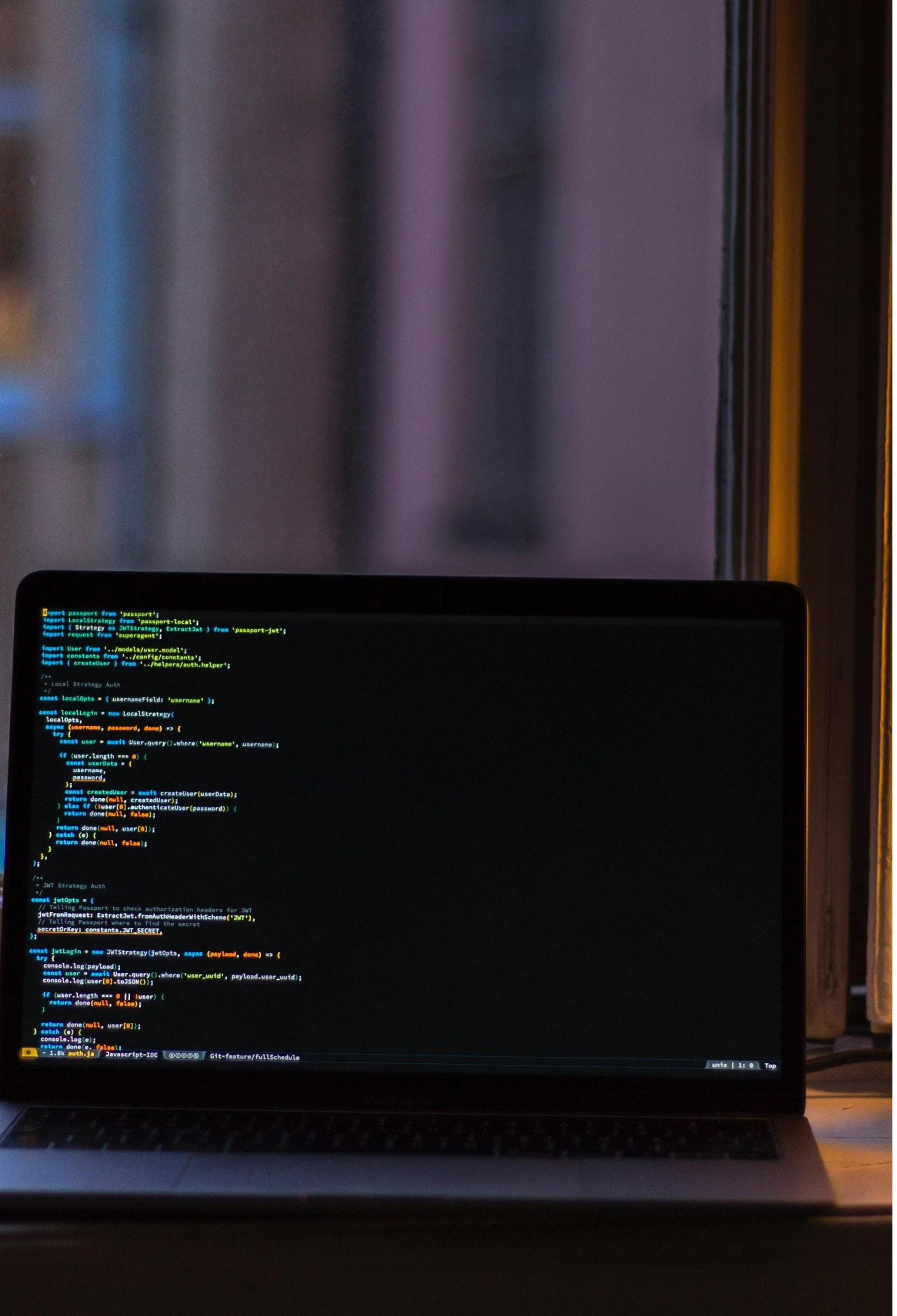
Eileen Cai

Data Scientist



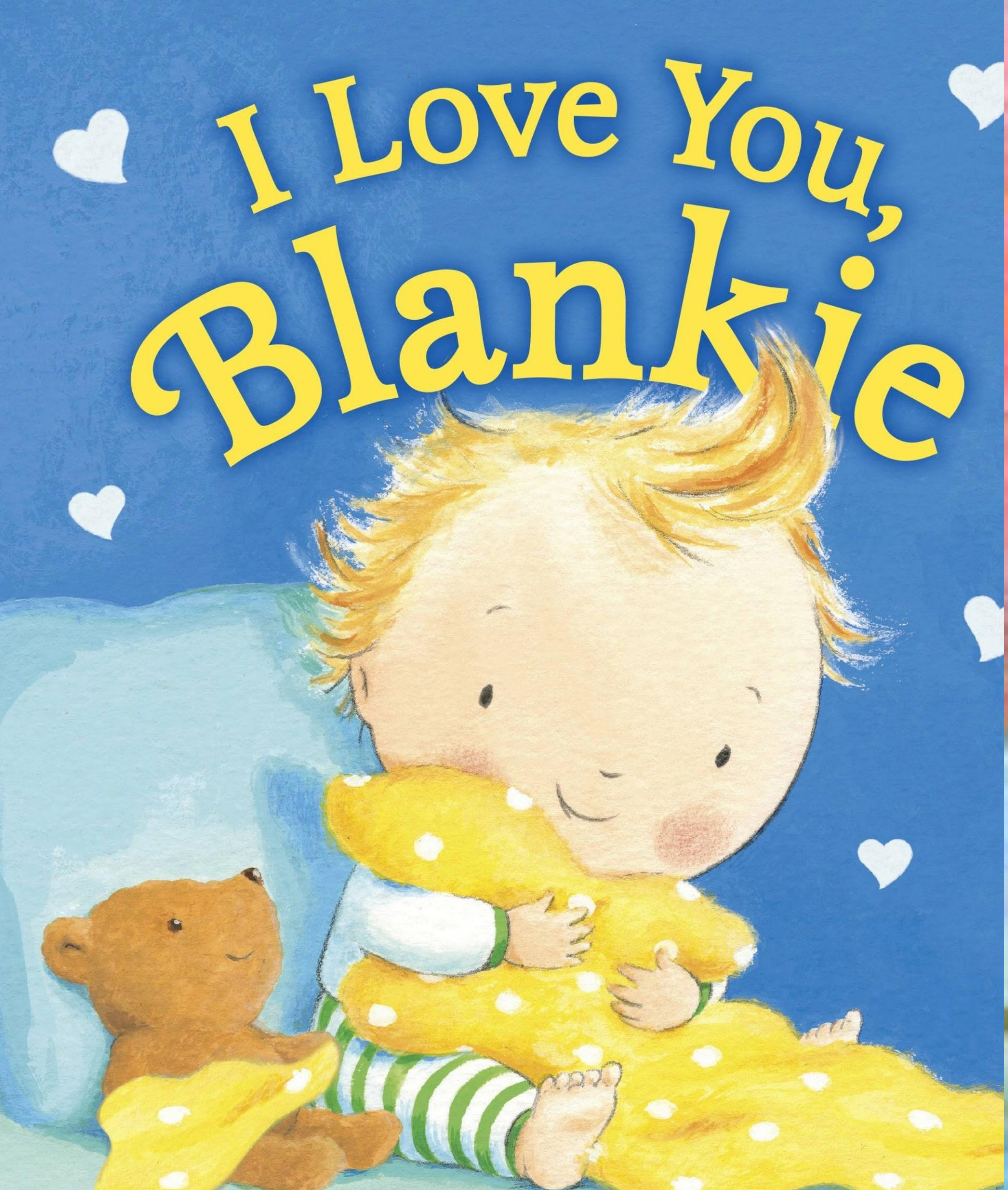
```
const events = [
  'dragenter',
  'dragleave',
  'dragover', // to allow drop
  'drop'
];
events.forEach(e => {
  fileDropZone.addEventListener(e, (ev) =
    ev.preventDefault();
    if (ev.type === 'dragenter') {
      fileDropZone.classList.add('soli
    }
    if (ev.type === 'dragleave') {
      fileDropZone.classList.remov
    }
    if(ev.type === 'drop') {
      fileDropZone.classList.re
      handleFiles(ev.dataTrans
      .then(values => valu
        tag.setAttribute(
        fileDropZone.app
```

Let’s make social media platforms safe!



```
import passport from 'passport';
import LocalStrategy from 'passport-local';
import { Strategy as JWTStrategy, ExtractJwt } from 'passport-jwt';
import request from 'superagent';

import User from '../models/user.model';
import constants from '../config/constants';
import { createUser } from '../helpers/auth.helper';

/**
 * Local Strategy Auth
 */
const localOpts = { usernameField: 'username' };

const localLogin = new LocalStrategy(
  localOpts,
  async (username, password, done) => {
    try {
      const user = await User.query().where('username', username);

      if (user.length === 0) {
        const userData = {
          username,
          password,
        };
        const createdUser = await createUser(userData);
        return done(null, createdUser);
      } else if (!user[0].authenticateUser(password)) {
        return done(null, false);
      }
      return done(null, user[0]);
    } catch (e) {
      return done(null, false);
    }
  },
);

/**
 * JWT Strategy Auth
 */
const jwtOpts = {
  // Telling Passport to check authorization headers for JWT
  jwtFromRequest: ExtractJwt.fromAuthHeaderWithScheme('JWT'),
  // Telling Passport where to find the secret
  secretOrKey: constants.JWT_SECRET,
};

const jwtLogin = new JWTStrategy(jwtOpts, async (payload, done) => {
  try {
    console.log(payload);
    const user = await User.query().where('user_uuid', payload.user_uuid);
    console.log(user[0].toJSON());

    if (user.length === 0 || !user) {
      return done(null, false);
    }

    return done(null, user[0]);
  } catch (e) {
    console.log(e);
    return done(e, false);
 0  - 1.8k auth.js  Javascript-IDE  ①②③④⑤  Git-feature/fullSchedule          unix | 1: 0  Top
```

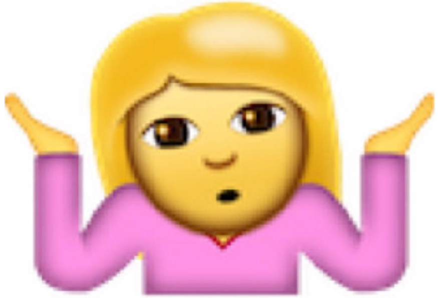
But sometimes it's
gibberish to me, too…


I Love You,
Blankie

Dataset (tweets)

Hate Speech (class 0)

Offensive Language (class 1)



Neither (class 2)

Next Steps

Data Processing

- Explore custom tokenization
- Explore Lemmatization

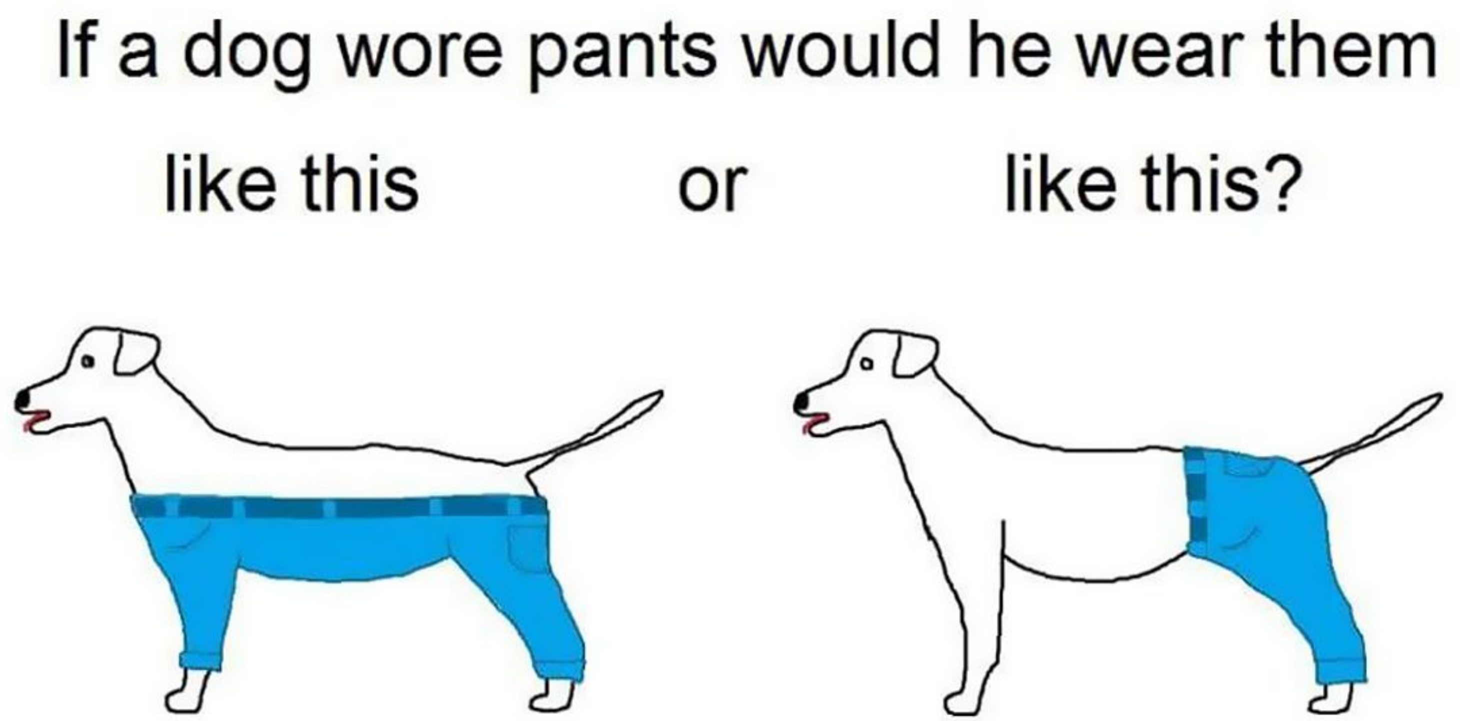
Inputs/Vectorization

- Larger set of data, retrain (bigger vocab)
- Stopwords for code, intent

Model Architecture (Try different models)

- Logistic Regression
- KNN
- Decision Tree

Questions?



If a dog wore pants would he wear them

like this or like this?