WINE!

Web Interaction & Visualization

TEAM 4: Lizzy, Ellen, Nii, Ryan, Andrew

TABLE OF CONTENTS

01. THE DATA

Kaggle

129,971

Wine Reviews (2018) from Kaggle

ZACKTHOUTT · UPDATED 4 YEARS AGO

3300 | New Notebook | Download (53 MB)

Wine Reviews

130k wine reviews with variety, location, winery, price, and description

Data | Code (3548) | Discussion (29) | Metadata

About Dataset

Context

After watching Somm (a documentary on master sommeliers) I wondered how I could create a predictive model to identify wines through blind tasting like a master sommelier would. The first step in this journey was gathering some data to train a model. I plan to use deep learning to predict the wine variety using words in the description/review. The model still won't be able to taste the wine, but theoretically it could identify the wine based on a description that a sommelier could give. If anyone has any ideas on how to accomplish this, please post them!

Usability ⓘ
7.94

License
CC BY-NC-SA 4.0

Expected update frequency
Not specified

winemag-data-130k-v2.json (79.28 MB)

This preview is truncated due to the large file size. The number of JSON items and individual items might be might be truncated. Create a Notebook or download this file to see the full content.

Download | Create Notebook

```
"root" :  171 items
  [ 0 - 100 ]
  [ 100 - 171 ]
```

Data Explorer

Version 4 (181.97 MB)

- winemag-data-130k-v2.csv
- winemag-data-130k-v2.json
- winemag-data_first150k.csv

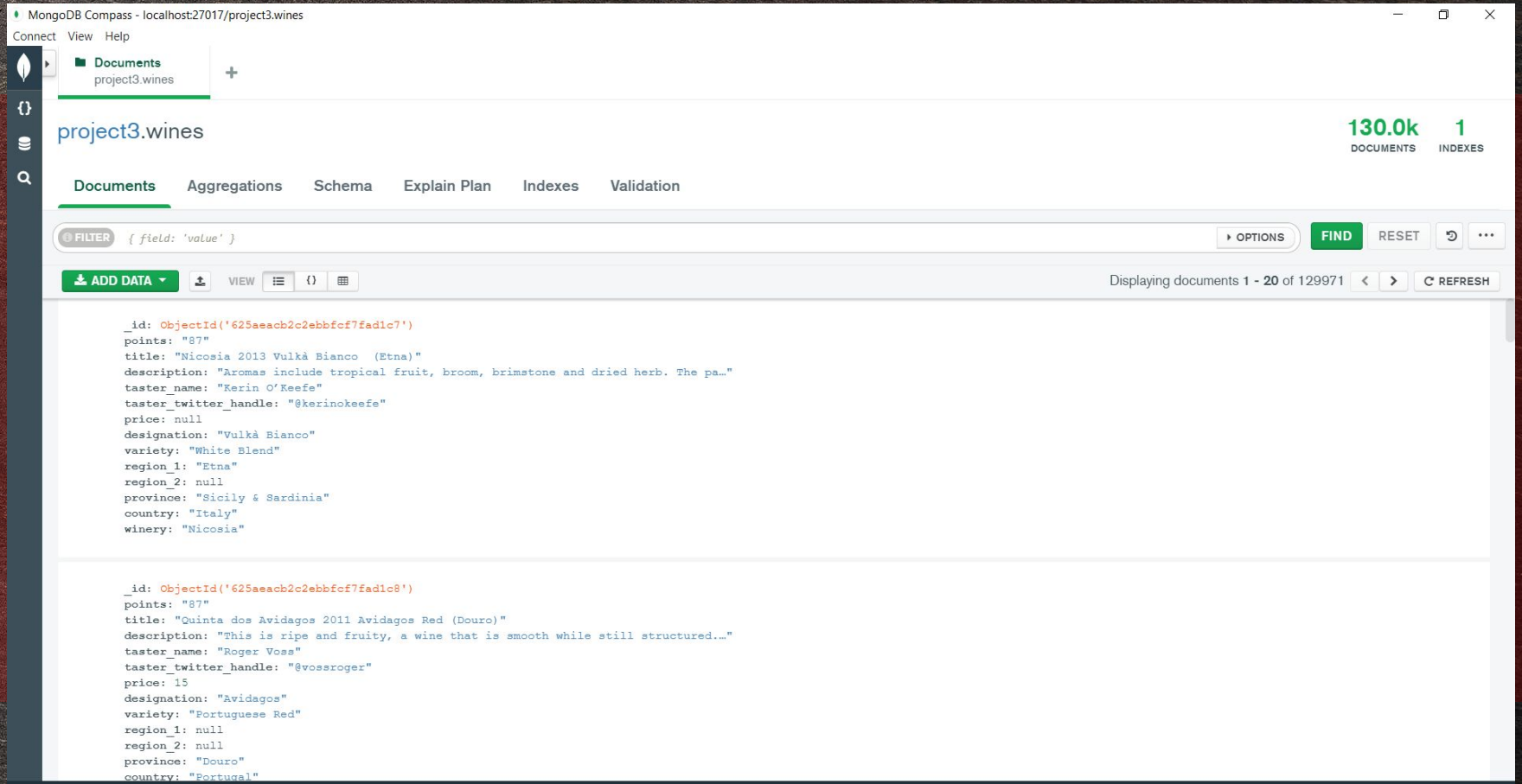
Summary

- 3 files
- 25 columns

02. FLASK APP & STORAGE

MongoDB & D3 Connection

MONGODB


MongoDB Compass - localhost:27017/project3.wines
Connect View Help
Documents
project3.wines
project3.wines
130.0k
DOCUMENTS
1
INDEXES
Documents
Aggregations
Schema
Explain Plan
Indexes
Validation
FILTER
{ field: 'value' }
OPTIONS
FIND
RESET
ADD DATA
VIEW
Displaying documents 1 - 20 of 129971
REFRESH
_id: ObjectId('625aeacb2c2ebbfcf7fad1c7')
points: "87"
title: "Nicosia 2013 Vulkà Bianco (Etna)"
description: "Aromas include tropical fruit, broom, brimstone and dried herb. The pa…"
taster_name: "Kerin O'Keefe"
taster_twitter_handle: "@kerinokeefe"
price: null
designation: "Vulkà Bianco"
variety: "White Blend"
region_1: "Etna"
region_2: null
province: "Sicily & Sardinia"
country: "Italy"
winery: "Nicosia"
_id: ObjectId('625aeacb2c2ebbfcf7fad1c8')
points: "87"
title: "Quinta dos Avidagos 2011 Avidagos Red (Douro)"
description: "This is ripe and fruity, a wine that is smooth while still structured.…"
taster_name: "Roger Voss"
taster_twitter_handle: "@vossroger"
price: 15
designation: "Avidagos"
variety: "Portuguese Red"
region_1: null
region_2: null
province: "Douro"
country: "Portugal"

FLASK MONGO CONNECTION

```
app = Flask(__name__)
CORS(app)

# Use PyMongo to establish Mongo connection
mongo = PyMongo(app, uri="mongodb://localhost:27017/project3")


@app.route("/")
def homepage():
    # want country, province, price, and points to be NOT null
    data = mongo.db.wines.find({},{'_id': 0,'taster_name':0,'taster_twitter_handle':0,
    'designation':0,'region_1':0,'region_2':0}).limit(5000)
    list_cur = list(data)
    json_data = jsonify(json_util.dumps(list_cur))
    return json_data


@app.route("/variety-list")
def variety_list():
    data = mongo.db.wines.distinct('variety')
    list_cur = list(data)
    return jsonify(list_cur)

@app.route("/select/<variety>")
def select(variety):
    data = mongo.db.wines.find({'variety':variety},{'_id': 0,'taster_name':0,
    'taster_twitter_handle':0,'designation':0,'region_1':0,'region_2':0})
    list_cur = list(data)
    return jsonify(list_cur)

@app.route("/coords")
def coords():
    data = mongo.db.merged.find()
    list_cur = list(data)
    json_data = jsonify(json_util.dumps(list_cur))
    return json_data


if __name__ == '__main__':
    app.run(debug=True)
```

CALLING THE FLASK APP IN JAVASCRIPT

```
function init() {
  d3.json("http://127.0.0.1:5000/variety-list").then(data => {
      console.log("read samples");
      console.log("data",data)
```

03. DROPDOWNS & INFO

Variables / Breakdown

WINE VARIETY DROPDOWN

```
function init() {
  d3.json("http://127.0.0.1:5000/variety-list").then(data => {
    console.log("read samples");
    console.log("data", data);
    let dropdownMenu = d3.select("#selDataset");
    data.forEach((uniqueVarietyList) => {
      dropdownMenu.append('option').text(uniqueVarietyList)
    })
    var result = data[0];
    buildMetadata(result)
  });
}
init();
```

TOP WINE INFO

```
function second_chart(sample) {

    d3.json(`http://127.0.0.1:5000/select/${sample}`).then(data => {
        console.log('top data');
        console.log(data)
        let results = data
        console.log(results)
        let top_results = results.slice(0, 3).reverse();
        console.log(top_results)
        let length = top_results.length
        console.log(length)

        //display the top option
        if (length > 1) {
            if (length > 1) {
                let displayinfo = d3.select('#sample-metadata2');
                displayinfo.html('');
                Object.entries(top_results[0]).forEach(k => {
                    console.log(k)
                    // displayinfo.append('panel-body').text(k[0] + ':' + k[1]).append('br');
                    displayinfo.append('panel-body').text(k[0].toUpperCase() + ': ' + k[1]).append('br');
                });
                //display the second top option
                let displayinfo2 = d3.select('#sample-metadata3');
                displayinfo2.html('');
                Object.entries(top_results[1]).forEach(k2 => {
                    console.log(k2)
                    displayinfo2.append('panel-body').text(k2[0].toUpperCase() + ':' + k2[1]).append('br');
                });
            } else {
                displayinfo2.append('panel-body').text(k2[0].toUpperCase() + ': ' + k2[1]).append('br');
            }
        } else {
                let displayinfo = d3.select('#sample-metadata2');
                displayinfo.html('');
                Object.entries(top_results[0]).forEach(k => {
                    console.log(k)
                    // displayinfo.append('panel-body').text(k[0] + ':' + k[1]).append('br');
                    displayinfo.append('panel-body').text(k[0].toUpperCase() + ': ' + k[1]).append('br');
                });
                let displayinfo2 = d3.select('#sample-metadata3');
                displayinfo2.html('');
            };
            //Display the top 10 options
            let title = data.filter(meta => meta.variety === sample);
            console.log(title)
            var top_ten = []
            for (var i = 0; i < title.length; i++) {
                title[i].title
                top_ten.push(title[i].title);
            }
            console.log(top_ten);
            number = [1,2,3,4,5,6,7,8,9,10]
            let top_ten2 = top_ten.slice(0, 10).reverse();
            let displayinfo3 = d3.select('#sample-metadata4');
            displayinfo3.html('')
            console.log(top_ten2)
            Object.entries(top_ten2).forEach(([key, value]) => {
                displayinfo3.append("panel-body").text(number[key] + `. `+ `${value}`).append('br');
            });
            console.log(top_ten2)
        });
    }
```

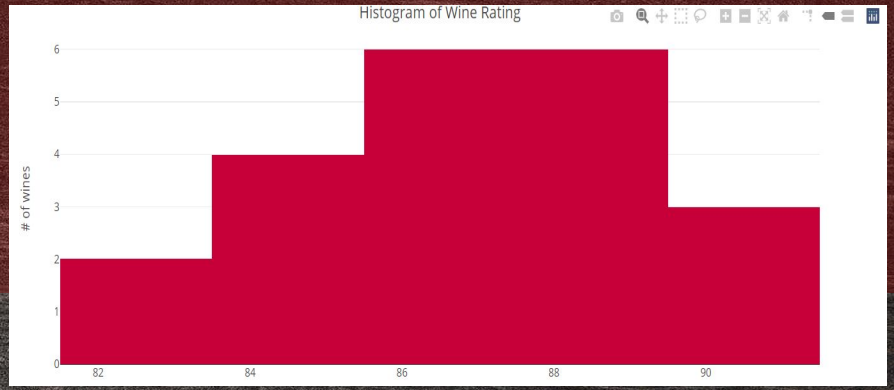
WINE VARIETY INFO

```
//Display the information based on which wine is selected
function second_chart(sample) {
    //read the flask
    d3.json("http://127.0.0.1:5000").then(data => {
        console.log('top data');
        var data = JSON.parse(data)
        console.log(data)
        // filter the data by variety
        let results = data.filter(varty => varty.variety === sample);
        console.log(results)
        //get the top 3 results
        let top_results = results.slice(0,3).reverse();
        console.log(top_results)
        let length = top_results.length
        console.log(length)

        //display the top option
        if (length > 1){
            let displayinfo = d3.select('#sample-metadata2');
            displayinfo.html('');
            Object.entries(top_results[0]).forEach(k =>{
                console.log(k)
                displayinfo.append('panel-body').text(k[0].toUpperCase()+': ' + k[1]).append('br');
            });
        //display the second top option
            let displayinfo2 = d3.select('#sample-metadata3');
            displayinfo2.html('');
            Object.entries(top_results[1]).forEach(k2 =>{
                console.log(k2)
                displayinfo2.append('panel-body').text(k2[0].toUpperCase()+': ' + k2[1]).append('br');
            });
        }else{
            let displayinfo = d3.select('#sample-metadata2');
            displayinfo.html('');
            Object.entries(top_results[0]).forEach(k =>{
                console.log(k)
                displayinfo.append('panel-body').text(k[0].toUpperCase()+': ' + k[1]).append('br');
            });
            let displayinfo2 = d3.select('#sample-metadata3');
            displayinfo2.html('');
        };
    });
}
```

04. VISUALIZATIONS

Charts / Graphs

BAR & BUBBLE GRAPH




```
74
75      let trace2 = {
76        x: (winepoints),
77        y: (wineprice),
78        //text: (otu_labelss),
79        mode: 'markers',
80        marker: {
81        color: (winepoints),
82        // size: (5),
83        colorscale: 'Fire',
84        text: [winename]
85      }
86    };
87
88    let bubbleLayout= {
89        title: 'Variety Price by Ratings',
90        margin: {t:0},
91        hovermode: "closest",
92        xaxis: {title: 'Wine Ratings'},
93        yaxis: {title:'Wine Prices'},
94        margin: {t:30}
95    };
96
97    let data2=[trace2]
98    Plotly.newPlot('bubble', data2, bubbleLayout);
99
100   var data = [
101     {
102       x: winepoints,
103       type: 'histogram',
104     marker: {
105       color: '#C70039',
106     },
107     }
108   ];
109   let barLayout= {
110     title: 'Histogram of Wine Rating',
111     margin: {t:0},
112     hovermode: "closest",
113     xaxis: {title: 'Wine Ratings'},
114     yaxis: {title:'# of wines'},
115     margin: {t:30}
116   };
117   Plotly.newPlot('bar', data,barLayout);
118 });
119 }
```


LEAFLET MAP

GEOCODING API

Concatenate winery and region to create an “address” and pull latitude and longitude of valid U.S. wineries using Google Maps Javascript API

```
const sleep = ms => new Promise(r => setTimeout(r, ms));

const jsonData = []
d3.json("http://127.0.0.1:5000").then(async data => {
    console.log("read samples");
    var mydata = JSON.parse(data)
    })
    // list of regions
    var regionList = [];
    for (var i = 0; i < mydata.length; i++) {
        regionList.push(mydata[i].region_1)
    }
    // list of wineries
    var wineryList = [];
    for (var i = 0; i < mydata.length; i++) {
        wineryList.push(mydata[i].winery)
    }
    // for coordinates
    var countryProv = wineryList.map(function (e, i,) {
        return [e, regionList[i]];
    }).map(([winery, region]) => `${winery}, ${region}`)

    var countryProvSet = new Set(countryProv)
    console.log(countryProvSet);
    countryProv = [...countryProvSet]
    console.log(countryProv);

    for (let i = 0; i < countryProv.length; i++) {
        var country = countryProv[i]
        var address = country
        console.log(address)
        var geocoder = new google.maps.Geocoder();
        try {
            await sleep(3000);
        var { results } = await geocoder.geocode({ address: address })
```

```
        jsonData.push({
            address: address,
            longitude: results[0].geometry.location.lng(),
            latitute: results[0].geometry.location.lat(),
        })

    } catch (e) {console.log(e)}


    console.log(jsonData)
    var json = JSON.stringify(jsonData);
    var blob = new Blob([json], { type: "application/json" });
    var url = URL.createObjectURL(blob);

      var a = document.createElement('a');
      a.download    = "backup.json";
      a.href        = url;
      a.textContent = "coords.json";
      a.click()}
```

Download coordinates and addresses as JSON file

PANDAS

Merge original collection using Pandas “on address” with new coordinate collection to include all winery information and its coordinates

In [61]:

```
client = MongoClient()
db = client.project3
wineryCoords = db.coordinates2.find()
list_cur1 = list(wineryCoords)
coords = pd.DataFrame(data = list_cur1)
wineInfo = db.wines.find("country" == "US")
list_cur2 = list(wineInfo)
info = pd.DataFrame(data = list_cur2)
info["address"] = info["winery"] + "," + " " + info["region_1"]
merged = info.merge(coords, on = "address")
merged['points'] = merged['points'].astype(int)
merged.drop(labels = ["taster_name", "taster_twitter_handle", "region_2"], axis = 1, inplace = True)
x = merged.groupby("address").mean().round(2)
```

In [63]:

```
x
```

Out[63]:

address	points	price	longitude	latitute
2Plank, Amador County	88.00	42.00	-120.77	38.35
Acrobat, Oregon	88.00	15.64	-120.55	43.80
Adelsheim, Dundee Hills	92.75	102.50	-123.05	45.34
Albatross Ridge, Carmel Valley	92.71	52.86	-121.73	36.48
Alleromb, Columbia Valley (WA)	90.62	62.77	-118.37	46.06
...	...	...	...	...
Wölffer, Long Island	88.65	27.84	-72.28	40.95
Y Rousseau, Mount Veeder	89.27	57.18	-122.45	38.38
Yardstick, Napa Valley	87.67	26.67	-122.27	38.50
Z'IVO, Eola-Amity Hills	87.50	35.00	-123.12	45.13
Zerba Cellars, Walla Walla Valley (WA)	86.20	29.60	-118.39	46.00

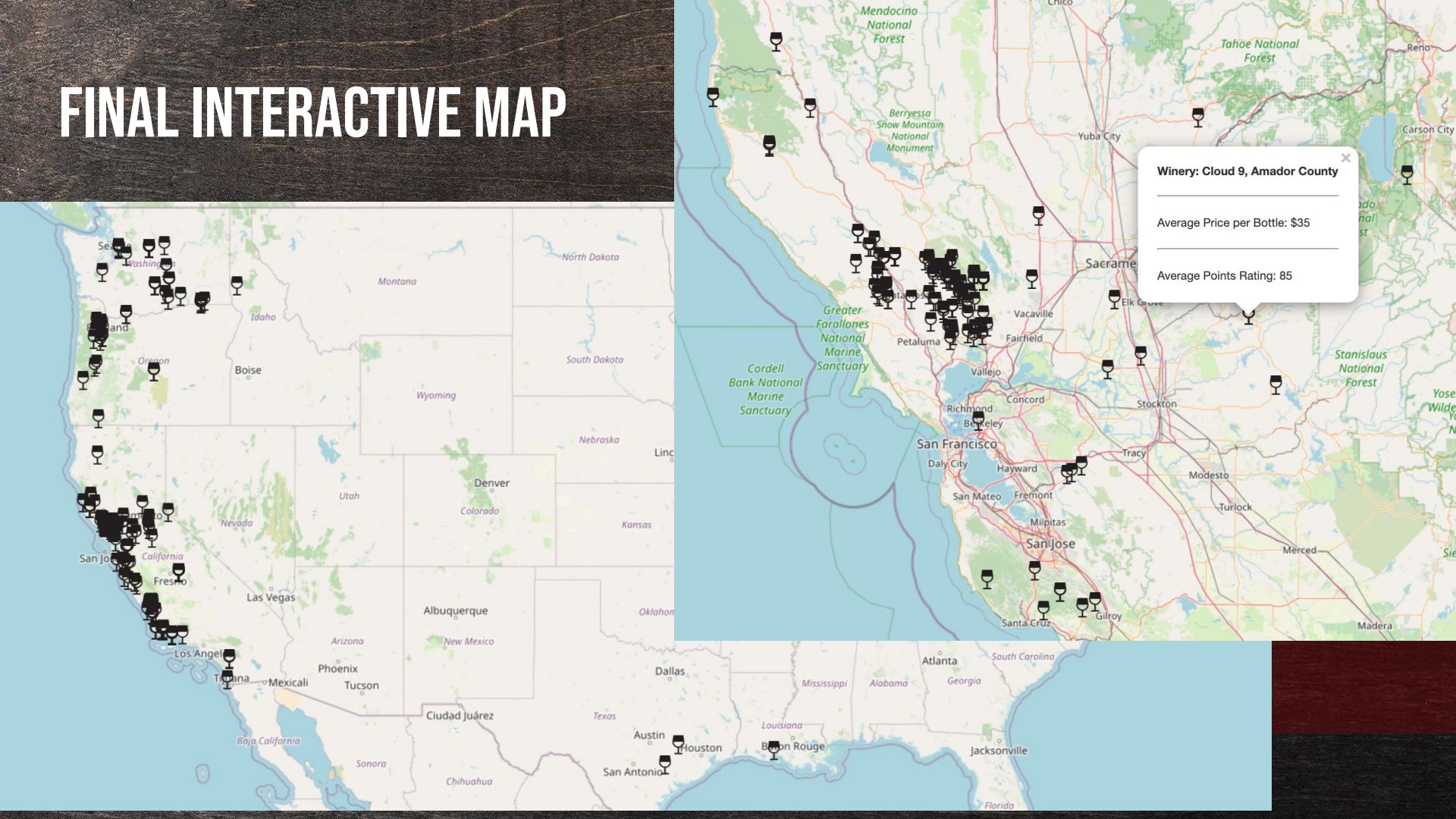
333 rows × 4 columns

MONGODB → LEAFLET

```
d3.json("http://127.0.0.1:5000/coords").then(async data => {
    console.log("read samples");
    var coorddata = JSON.parse(data)
    console.log(coorddata);

    var myMap = L.map("map", {
        center: [39.0997, -94.5786],
        zoom: 4.4
    });

    L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
        attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors'
    }).addTo(myMap);
    var markers = []
    var lats = []
    var lngs = []
    var winery = []
    var avgPoints = []
    var avgPrice = []

    for (const [key, value] of Object.entries(coorddata[0].latitute)) {
        lats.push(`${value}`);
        winery.push(`${key}`);
    }
    console.log(winery)
    for (const [key, value] of Object.entries(coorddata[0].longitude)) {
        lngs.push(`${value}`);
    }
    for (const [key, value] of Object.entries(coorddata[0].points)) {
        avgPoints.push(`${value}`);
    }
    for (const [key, value] of Object.entries(coorddata[0].price)) {
        avgPrice.push(`${value}`);
    }
    var myIcon = L.icon({
        iconUrl: '/64x64.png',
        iconSize: [20, 20],
    });
    for (var i = 0; i < lats.length; i++) {
        var marker = L.marker([lats[i], lngs[i]], { icon: myIcon }).bindPopup("<h4>Winery: " + winery[i] +
            "</h4><hr><p>Average Price per Bottle: $" + avgPrice[i] +
            "</p><hr><p>Average Points Rating: " + avgPoints[i]).openPopup();
        markers.push(marker);
    }
    for (const [key, value] of Object.entries(coorddata[0].points)) {
        avgPoints.push(`${value}`);
    }
    for (const [key, value] of Object.entries(coorddata[0].price)) {
        avgPrice.push(`${value}`);
    }
    for (var i = 0; i < 333; i++) {
        markers.push(marker);
    };
    var newlayerGroup = L.layerGroup(markers)
    newlayerGroup.addTo(myMap)
})
```

Export merged DataFrame as JSON to MongoDB to use in Leaflet script to plot wineries and pop up info

FINAL INTERACTIVE MAP



05. HTML/CSS

Web Design

HTML

Bootstrap 5.0.2

- Slideshow/Carousel

```
<!-- Carousel -->
<div id="demo" class="carousel slide" data-bs-ride="carousel">

  <!-- Indicators/dots -->
  <div class="carousel-indicators">
    <button type="button" data-bs-target="#demo" data-bs-slide-to="0" class="active"></button>
    <button type="button" data-bs-target="#demo" data-bs-slide-to="1"></button>
    <button type="button" data-bs-target="#demo" data-bs-slide-to="2"></button>
  </div>


<!-- The slideshow/carousel -->
<div class="carousel-inner">
  <div class="carousel-item active">
    <div class="carousel-caption" style="background-color: rgba(0, 0, 0, 0.6);">
        <h1>Wine a Little, Laugh a Lot</h1>
        <!-- <p>We had such a great time in LA!</p> -->
        <a class="btn btn-outline-light btn-lg m-2" href="mapindex.html"
        role="button" rel="nofollow" target="_blank">MAPS</a>
    </div>
  </div>
  <div class="carousel-item">
    <div class="carousel-caption" style="background-color: rgba(0, 0, 0, 0.6);">
      <h1>There's Always Time for a Glass of Wine</h1>
      <!-- <p>Thank you, Chicago!</p> -->
      <a class="btn btn-outline-light btn-lg m-2" href="mapindex.html"
      role="button" rel="nofollow" target="_blank">MAPS</a>
    </div>
  </div>
  <div class="carousel-item">
    <div class="carousel-caption" style="background-color: rgba(0, 0, 0, 0.6);">
      <h1>But First, Wine!</h1>
      <!-- <p>We love the Big Apple!</p> -->
      <a class="btn btn-outline-light btn-lg m-2" href="mapindex.html"
      role="button" rel="nofollow" target="_blank">MAPS</a>
    </div>
  </div>
</div>
```

THANK YOU

SOURCES

Kaggle

Slidesgo: Presentation template

Flaticon: icons

Freepik: infographics and images

MDBoostrap.com

Images:

- https://hdwallsource.com/img/2016/8/vineyard-wallpaper-background-51268-52964-hd-wallpapers.jpg
- https://jaksonhospitality.com/blog/wp-content/uploads/2017/09/vineyard-grapes-home-slideshow.jpg
- https://mdbootstrap.com/img/Photos/Others/images/78.jpg
- https://envato-shoebox-0.imgix.net/e0fe/fbc7-16d5-4a95-a1f0-8955c5c23a3f/Wine001-5535.jpg?auto=compress%2Cformat&fit=max&mark=https%3A%2F%2Felements-assets.envato.com%2Fstatic%2Fwatermark2.png&markalign=center%2Cmiddle&markalpha=18&w=1600&s=368091e2e1ddd64938e13debe198bd9c
- https://images.pexels.com/photos/39511/purple-grapes-vineyard-napa-valley-napa-vineyard-39511.jpeg?auto=compress&cs=tinysrgb&dpr=1&w=500
- https://envato-shoebox-0.imgix.net/108b/3ace-31b8-46b5-96b1-2652e85b9f6c/Wine001-5638.jpg?auto=compress%2Cformat&fit=max&mark=https%3A%2F%2Felements-assets.envato.com%2Fstatic%2Fwatermark2.png&markalign=center%2Cmiddle&markalpha=18&w=1600&s=fd760a24d0147ecf45258a4b24a4923e
- https://images.creativemarket.com/0.1.0/ps/2810592/1360/907/m1/fpnw/wm1/ixvowhloyhcgiwm1wthk5aeipbwvnf0d34hph2ykvzswfdiopc3vr2nzwxtok8ys-.jpg?1496772611&s=98c8a1beafe05214de1cd7fe7a0febad