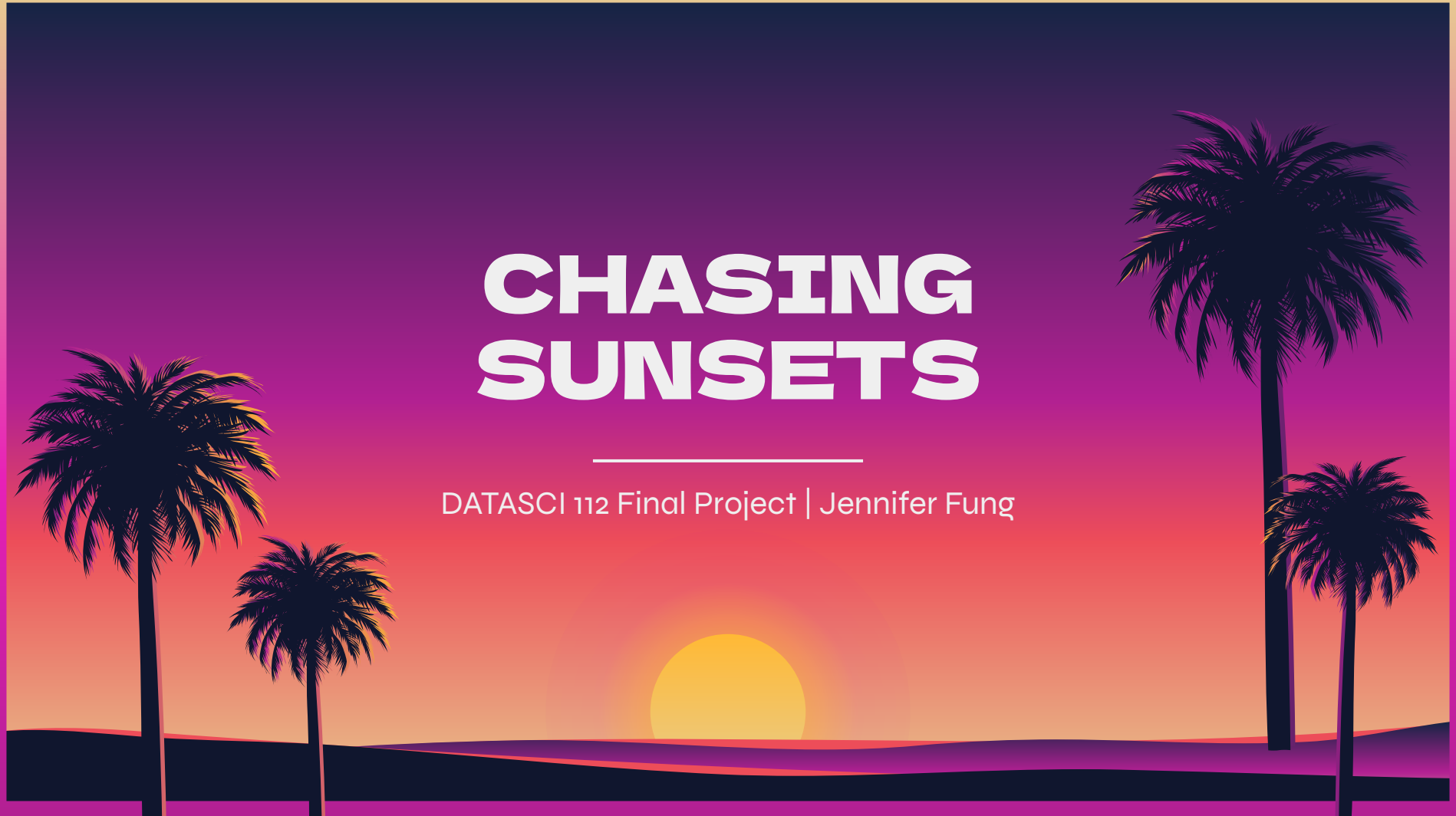

CHASING
SUNSETS
DATASCI 112 Final Project | Jennifer Fung

TABLE OF CONTENTS

01

RESEARCH MOTIVATION

02

DATA COLLECTION AND CLEANING

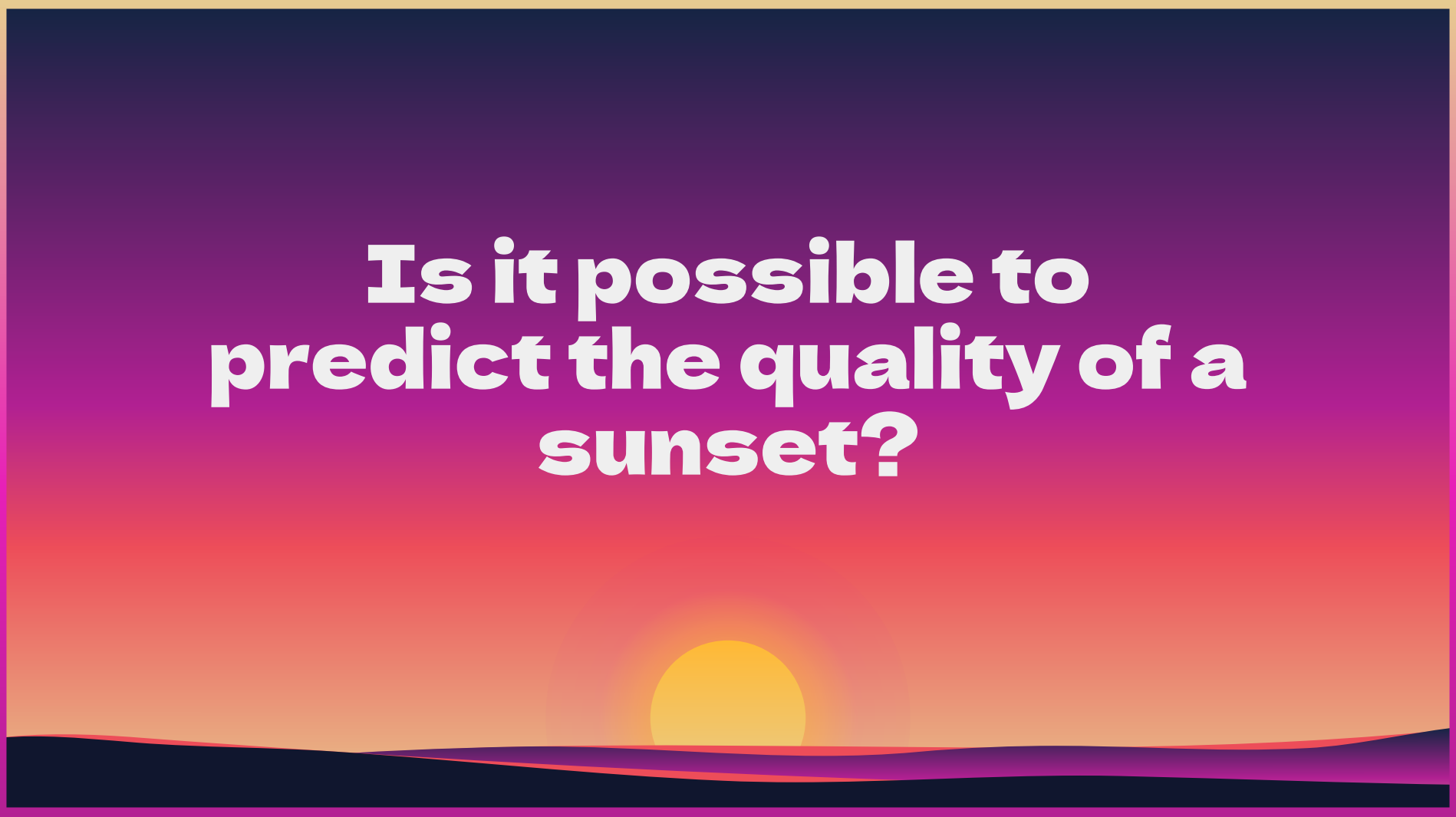
03

DATA VISUALIZATION

04

MACHINE LEARNING

Is it possible to predict the quality of a sunset?



2130

sunsets in 10 cities every day over 7 months

LOS ANGELES, SAN FRANCISCO, SAN DIEGO, SEATTLE, CHICAGO, MIAMI, PHILADELPHIA, NYC, BOSTON, WASHINGTON DC

	Date	City	Raw Number of Posts	Residual After Controlling for Time	Good Sunset
Los_Angeles2015-11-01	2015-11-01	Los_Angeles	365	-0.5107030775772110	0.0
Los_Angeles2015-11-02	2015-11-02	Los_Angeles	625	0.4523763388675570	1.0
Los_Angeles2015-11-03	2015-11-03	Los_Angeles	610	0.4930561177835950	1.0
Los_Angeles2015-11-04	2015-11-04	Los_Angeles	505	0.23602293714068700	0.0
Los_Angeles2015-11-05	2015-11-05	Los_Angeles	376	-0.026923240153921600	0.0
Los_Angeles2015-11-06	2015-11-06	Los_Angeles	391	-0.08362969111648470	0.0
Los_Angeles2015-11-07	2015-11-07	Los_Angeles	45	-0.8295262280920170	0.0
Los_Angeles2015-11-08	2015-11-08	Los_Angeles	623	0.18667133909539700	0.0
Los_Angeles2015-11-09	2015-11-09	Los_Angeles	330	-0.18596569239174900	0.0
Los_Angeles2015-11-10	2015-11-10	Los_Angeles	333	-0.1057311005985000	0.0
Los_Angeles2015-11-11	2015-11-11	Los_Angeles	360	-0.07072180408245570	0.0
Los_Angeles2015-11-12	2015-11-12	Los_Angeles	385	0.007143898871248220	0.0
Los_Angeles2015-11-13	2015-11-13	Los_Angeles	318	-0.23159992621471300	0.0
Los_Angeles2015-11-14	2015-11-14	Los_Angeles	451	0.08325542285203370	0.0
Los_Angeles2015-11-16	2015-11-16	Los_Angeles	316	-0.20444477981039600	0.0
Los_Angeles2015-11-17	2015-11-17	Los_Angeles	347	-0.0626009924052332	0.0

	sunset	City	Date	Sunset Time
0	2015-11-01T18:01	Los Angeles	2015-11-01	18:01
1	2015-11-02T18:00	Los Angeles	2015-11-02	18:00
2	2015-11-03T17:59	Los Angeles	2015-11-03	17:59
3	2015-11-04T17:59	Los Angeles	2015-11-04	17:59
4	2015-11-05T17:58	Los Angeles	2015-11-05	17:58
...	...	...	...	...
208	2016-05-27T17:25	Washington DC	2016-05-27	17:25

WEATHER VARIABLES

CLOUDCOVER

OVERALL, LOW, MID, HIGH

RADIATION

SHORTWAVE, DIRECT, DIFFUSE, DIRECT NORMAL IRRADIANCE

WIND

WIND SPEED, WIND DIRECTION

HUMIDITY

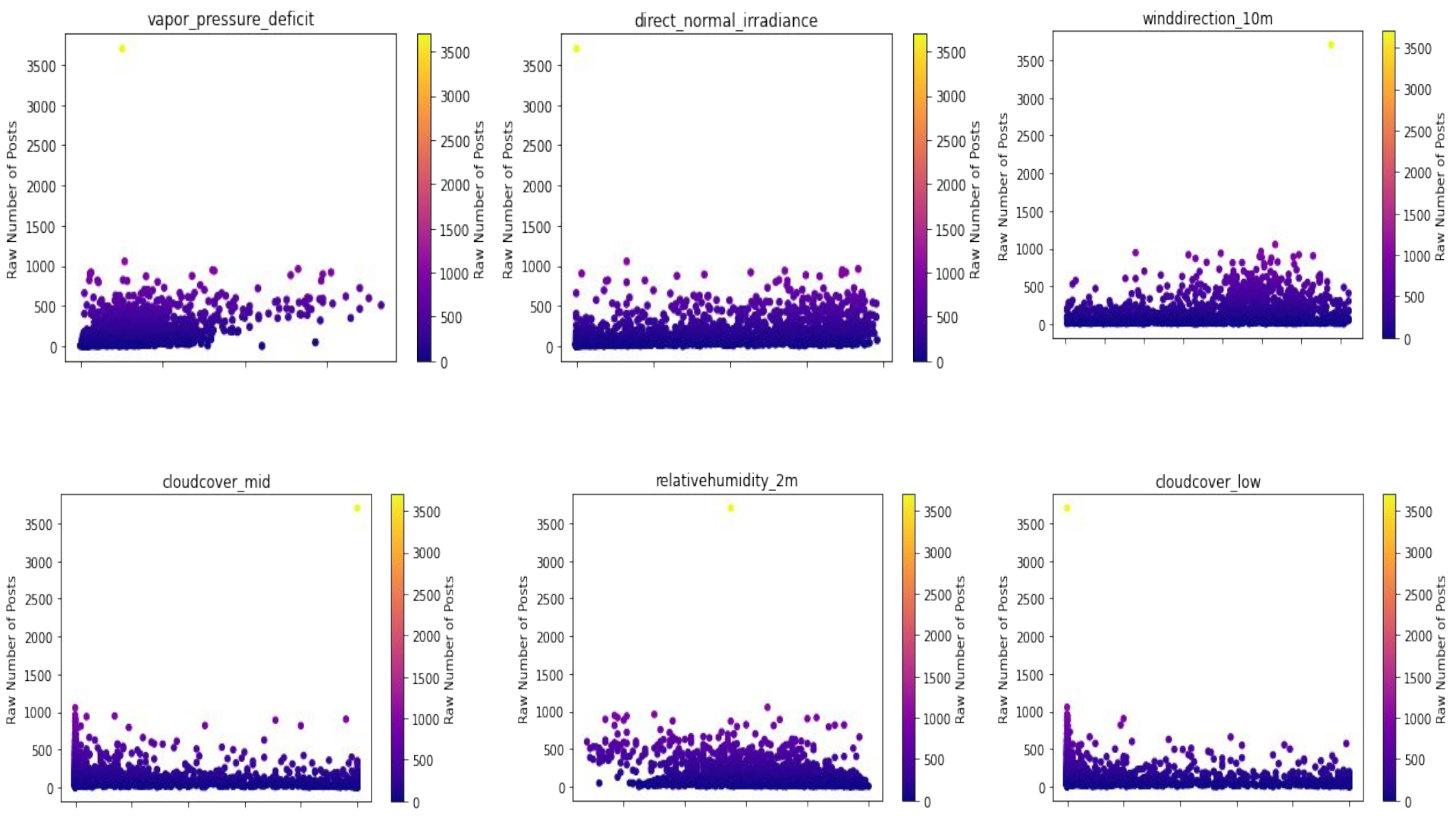
DEWPOINT

SURFACE PRESSURE

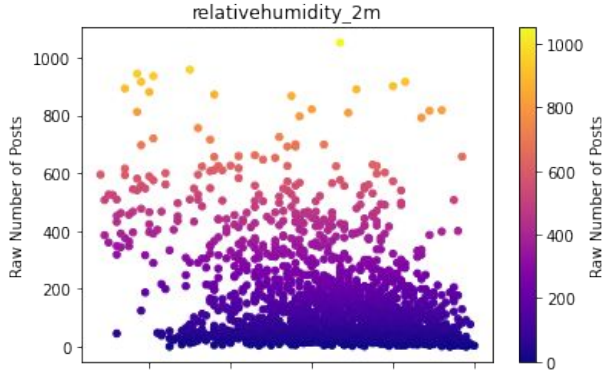
SURFACE PRESSURE, MEAN SEA LEVEL PRESSURE

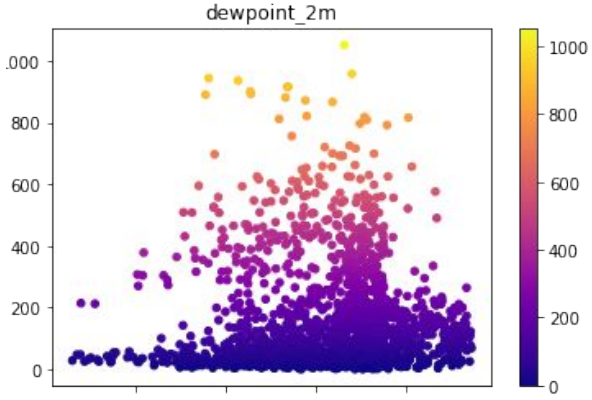
direct_normal_irradiance	windspeed_10m	winddirection_10m	et0_fao_evapotranspiration	vapor_pressure_deficit	Date	Hour	City
0.0	6.9	199	0.03	0.43	2015-11-01	00:00	Washington DC
0.0	7.2	196	0.03	0.43	2015-11-01	01:00	Washington DC
0.0	8.0	193	0.03	0.37	2015-11-01	02:00	Washington DC
0.0	8.3	199	0.02	0.35	2015-11-01	03:00	Washington DC
0.0	8.9	202	0.02	0.31	2015-11-01	04:00	Washington DC
...	...	...	...	...	...	...	...
655.6	6.0	211	0.17	1.33	2016-05-31	19:00	Los Angeles
342.7	5.6	214	0.06	0.91	2016-05-31	20:00	Los Angeles
0.0	4.2	212	0.02	0.66	2016-05-31	21:00	Los Angeles
0.0	3.0	197	0.00	0.53	2016-05-31	22:00	Los Angeles
0.0	3.2	172	0.00	0.45	2016-05-31	23:00	Los Angeles

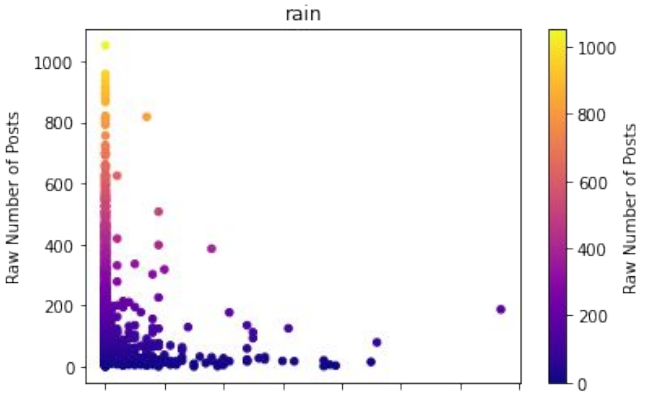
	City	Date	Good Sunset	Sunset Time	Raw Number of Posts	relativehumidity_2m	dewpoint_2m	pressure_msl	surface_pressure	rain	...	cloudcove
0	Los Angeles	2015-11-01	0.0	18:01	365	36	47.6	1013.4	996.7	0.0	...	
1	Los Angeles	2015-11-02	1.0	18:00	625	76	56.1	1007.8	990.8	0.2	...	
2	Los Angeles	2015-11-03	1.0	17:59	610	38	38.6	1008.9	991.9	0.0	...	
3	Los Angeles	2015-11-04	0.0	17:59	505	38	39.1	1013.8	996.8	0.0	...	
4	Los Angeles	2015-11-05	0.0	17:58	376	14	17.8	1019.5	1002.5	0.0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
1786	Seattle	2016-05-27	0.0	20:56	33	67	44.4	1019.6	1017.5	0.2	...	
1787	Seattle	2016-05-28	0.0	20:57	74	80	49.4	1016.6	1014.5	0.3	...	
1788	Seattle	2016-05-29	0.0	20:58	104	66	46.0	1022.1	1020.0	0.0	...	
1789	Seattle	2016-05-30	0.0	20:59	66	57	46.2	1021.9	1019.9	0.0	...	
1790	Seattle	2016-05-31	0.0	21:00	4	65	51.4	1013.6	1011.6	0.0	...	

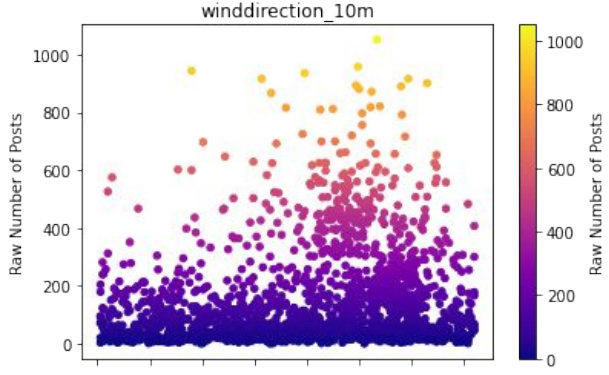

vapor_pressure_deficit
direct_normal_irradiance
winddirection_10m
cloudcover_mid
relativehumidity_2m
cloudcover_low
Raw Number of Posts
0
500
1000
1500
2000
2500
3000
3500

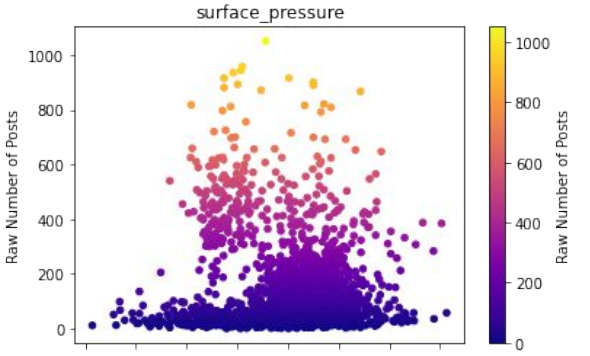

©Inga Sarda-Sorensen

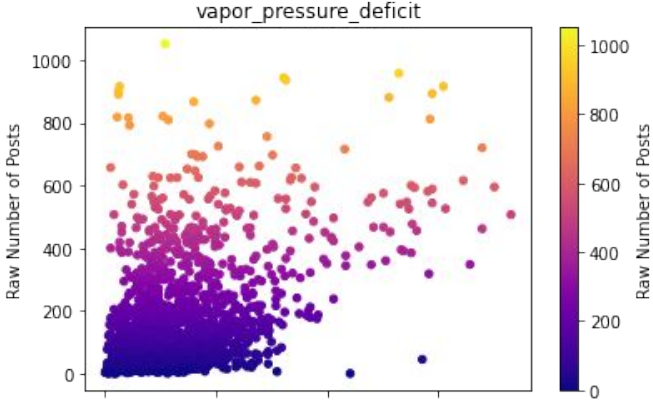

relativehumidity_2m
Raw Number of Posts
0
200
400
600
800
1000
Raw Number of Posts


dewpoint_2m
Raw Number of Posts
0
200
400
600
800
1000
Raw Number of Posts


rain
Raw Number of Posts
0
200
400
600
800
1000
Raw Number of Posts


winddirection_10m
Raw Number of Posts
0
200
400
600
800
1000
Raw Number of Posts


surface_pressure
Raw Number of Posts
0
200
400
600
800
1000
Raw Number of Posts


vapor_pressure_deficit
Raw Number of Posts
0
200
400
600
800
1000
Raw Number of Posts

MACHINE LEARNING

```
X = sunsetData[features]
y_bad = sunsetData["Raw Number of Posts"]

X_train, X_test, y_train, y_test = train_test_split(X, y_bad, test_size = .33)

scalers = [StandardScaler, MinMaxScaler, Normalizer]

for scaler in scalers:
  col_transformer = make_column_transformer((scaler(), features),
                                            remainder="drop")
  pipeline = make_pipeline(col_transformer, KNeighborsRegressor(n_neighbors=20, metric = "manhattan"))
  print(scaler, -cross_val_score(
      pipeline, X_train, y_train,
      scoring = "neg_mean_squared_error",
      cv=5).mean())
  pipeline.fit(X_train, y_train)
  y_pred = pipeline.predict(X_test)
```

```
<class 'sklearn.preprocessing._data.StandardScaler'> 19402.8119395659
<class 'sklearn.preprocessing._data.MinMaxScaler'> 19842.92236593445
<class 'sklearn.preprocessing._data.Normalizer'> 22776.927956685846
```

```
goodSunsets = sunsetData[sunsetData["Good Sunset"] == 1.0]
goodSunsets
```

	City	Date	Good Sunset	Sunset Time	Raw Number of Posts	relativehumidity_2m	dewpoint_2m	pressure_msl	surface_pressure	rain	...	cloudcover_mid	cloudcover_high
1	Los Angeles	2015-11-02	1.0	18:00	625	76	56.1	1007.8	990.8	0.2	...	67	0
2	Los Angeles	2015-11-03	1.0	17:59	610	38	38.6	1008.9	991.9	0.0	...	36	0
26	Los Angeles	2015-11-28	1.0	17:45	697	18	17.7	1016.0	998.8	0.0	...	0	0
35	Los Angeles	2015-12-07	1.0	17:45	893	14	25.7	1016.8	1000.1	0.0	...	0	0
36	Los Angeles	2015-12-08	1.0	17:45	812	17	32.0	1015.3	998.7	0.0	...	0	97
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1769	Seattle	2016-05-10	1.0	20:35	215	36	37.9	1020.6	1018.6	0.0	...	0	39
1770	Seattle	2016-05-11	1.0	20:37	155	35	40.6	1012.9	1010.9	0.0	...	0	0
1771	Seattle	2016-05-12	1.0	20:38	164	38	43.0	1014.9	1012.9	0.0	...	0	23
1772	Seattle	2016-05-13	1.0	20:39	174	50	53.2	1010.8	1008.8	0.0	...	46	8

```
goodSunsets = sunsetData[sunsetData["Good Sunset"] == 1.0]
goodSunsets
```

	City	Date	Good Sunset	Sunset Time	Raw Number of Posts	relativehumidity_2m	dewpoint_2m	pressure_msl	surface_pressure	rain	...	cloudcover_mid	cloudcover_high
1	Los Angeles	2015-11-02	1.0	18:00	625	76	56.1	1007.8	990.8	0.2	...	67	0
2	Los Angeles	2015-11-03	1.0	17:59	610	38	38.6	1008.9	991.9	0.0	...	36	0
26	Los Angeles	2015-11-28	1.0	17:45	697	18	17.7	1016.0	998.8	0.0	...	0	0
35	Los Angeles	2015-12-07	1.0	17:45	893	14	25.7	1016.8	1000.1	0.0	...	0	0
36	Los Angeles	2015-12-08	1.0	17:45	812	17	32.0	1015.3	998.7	0.0	...	0	97
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1769	Seattle	2016-05-10	1.0	20:35	215	36	37.9	1020.6	1018.6	0.0	...	0	39
1770	Seattle	2016-05-11	1.0	20:37	155	35	40.6	1012.9	1010.9	0.0	...	0	0
1771	Seattle	2016-05-12	1.0	20:38	164	38	43.0	1014.9	1012.9	0.0	...	0	23
1772	Seattle	2016-05-13	1.0	20:39	174	50	53.2	1010.8	1008.8	0.0	...	46	8

```
X = sunsetData[features]
y_bad = sunsetData["Raw Number of Posts"]
```

```
X = sunsetData[features]
y = sunsetData["Good Sunset"]
```

```
[68] from sklearn.ensemble import RandomForestClassifier
     from sklearn.metrics import accuracy_score
     from sklearn.model_selection import RandomizedSearchCV
     from sklearn.metrics import classification_report


     clf = RandomForestClassifier(n_estimators = 100)
     clf.fit(X_train, y_train)
     y_pred = clf.predict(X_test)
     accuracy = accuracy_score(y_test, y_pred)
     print("Accuracy:", accuracy)

     Accuracy: 0.8307952622673435
```

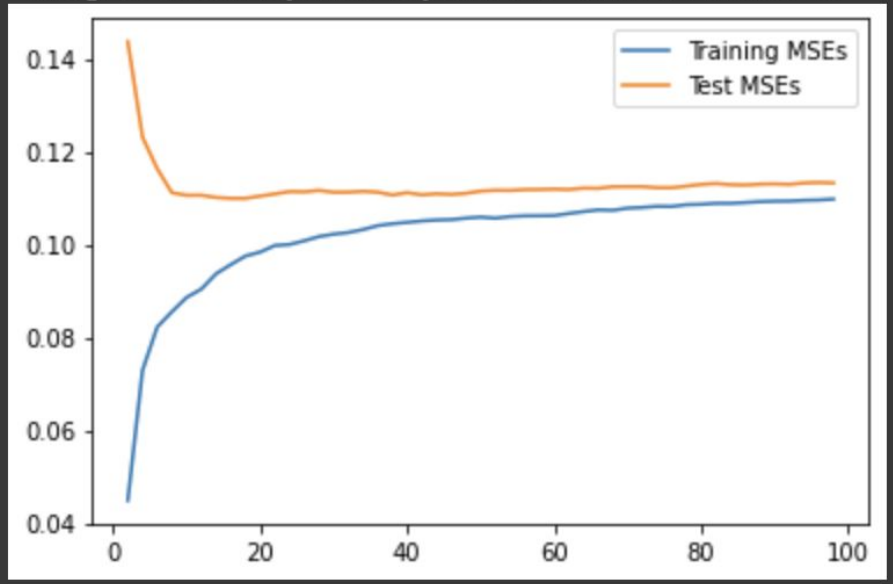
```
X = sunsetData[features]
y = sunsetData["Good Sunset"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = .33)

scalers = [StandardScaler, MinMaxScaler, Normalizer]

for scaler in scalers:
  col_transformer = make_column_transformer((scaler(), features),
                                            remainder="drop")
  pipeline = make_pipeline(col_transformer, KNeighborsRegressor(n_neighbors=20, metric = "manhattan"))
  print(scaler, -cross_val_score(
      pipeline, X_train, y_train,
      scoring = "neg_mean_squared_error",
      cv=5).mean())
  pipeline.fit(X_train, y_train)
```

```
<class 'sklearn.preprocessing._data.StandardScaler'> 0.113045850767085 07
<class 'sklearn.preprocessing._data.MinMaxScaler'> 0.11497058054393303
<class 'sklearn.preprocessing._data.Normalizer'> 0.11813473675034866
```

[63]

```
plt.plot(ks, train_mses, label = "Training MSEs")
plt.plot(ks, test_mses, label = "Test MSEs")
plt.legend(loc = "upper right")
```

```
<matplotlib.legend.Legend at 0x7f4214c66340>
```

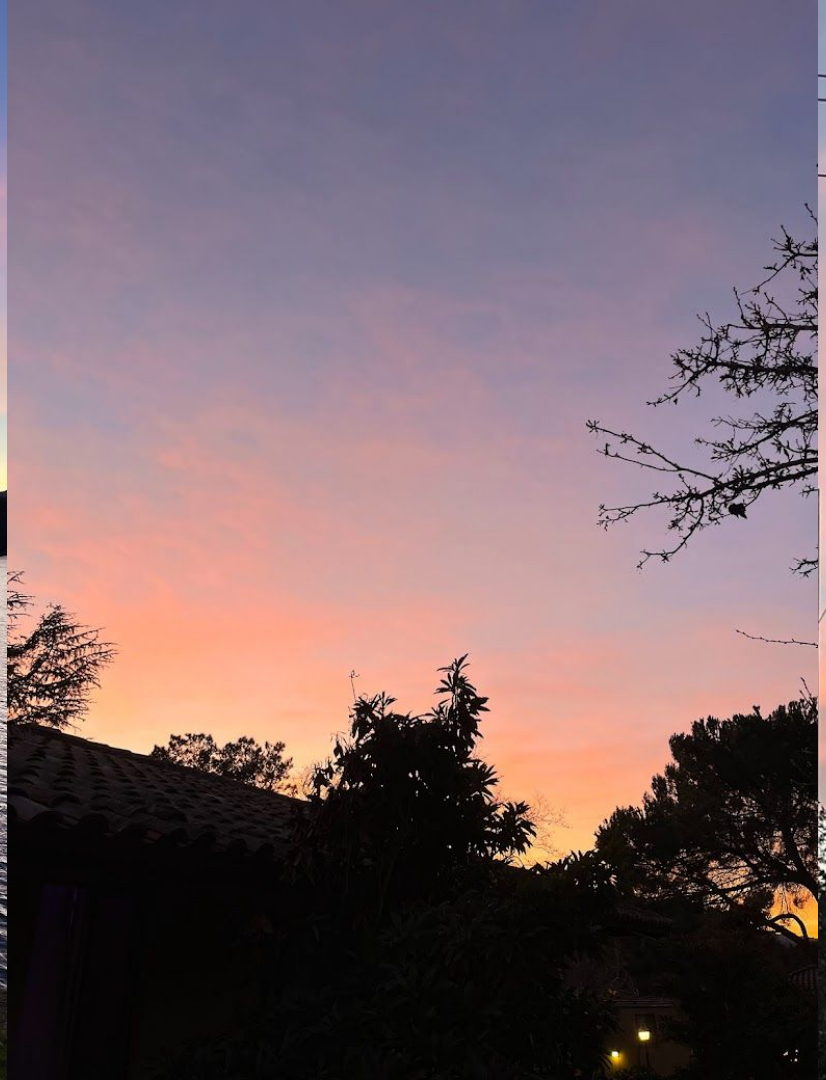


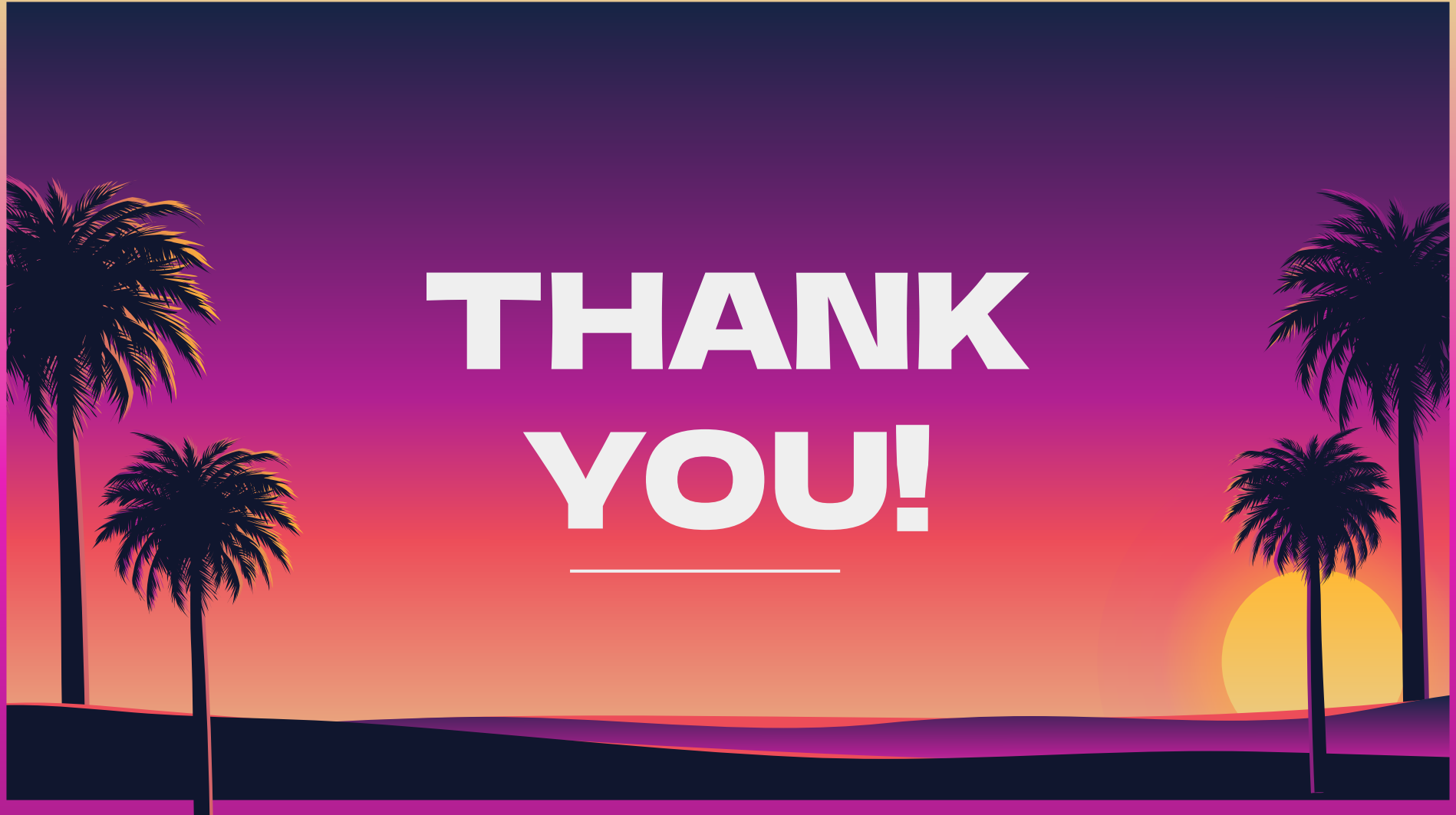

```
pd.Series(test_mses, index = ks).idxmin()
```

```
18
```

```
X = sunsetData[features]
y = sunsetData["Good Sunset"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = .33)

col_transformer = make_column_transformer((StandardScaler(), features),
                                          remainder="drop")
pipeline = make_pipeline(col_transformer, KNeighborsRegressor(n_neighbors=18, metr
print(StandardScaler, -cross_val_score(
      pipeline, X_train, y_train,
      scoring = "neg_mean_squared_error",
      cv=5).mean())
```

```
<class 'sklearn.preprocessing._data.StandardScaler'> 0.10710837551870792
```



THANK
YOU!