Team Terriers
Locational Power Grid Study

Ziming Hua, Wenxuan Fu

Zhanfeng Li, Haihan Yuan

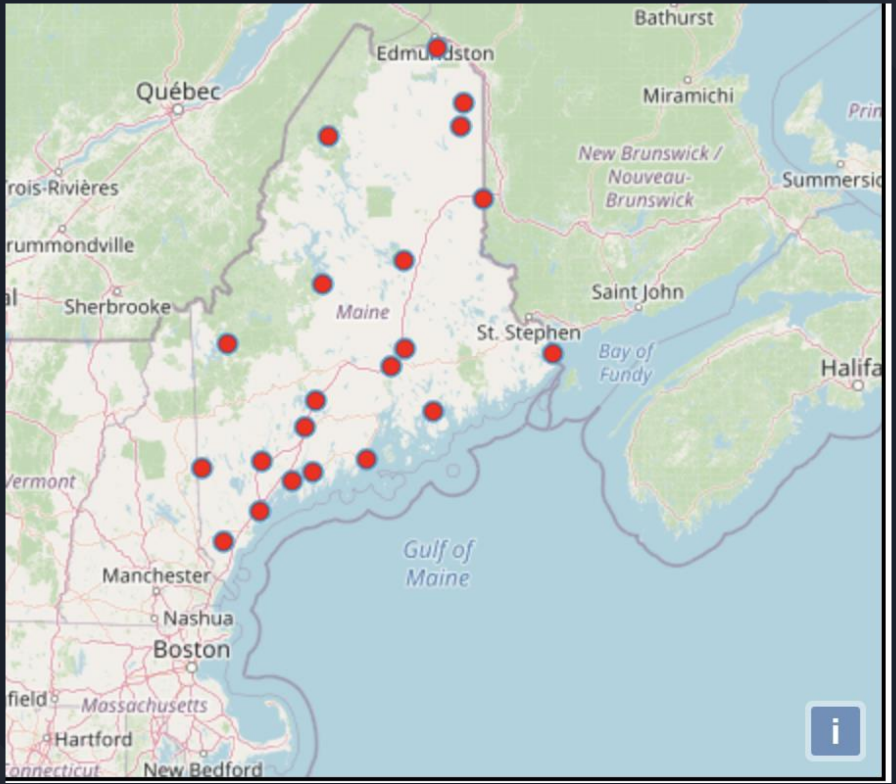
Central Maine Power

Given:

- Actual load data from 2019 to 2021 of CMP
- Historical weather data from Iowa State University

Goal:

- Predict the hourly loads of CMP for the next day



Cleaning Data

The original data we download from Iowa State University ASOS Network contains contains 30 variables. However, most of them are not that helpful and contains WAY TOO MUCH missing values.

After carefully reviewing each variables, we decided to keep the following variables:

'station', 'valid', 'tmpf', 'dwpf', 'relh', 'drct', 'sknt', 'p01i',

'alti', 'mslp', 'vsby','skyc1','skyl1','feel'

```
> freq.na(asos2019_2022_routine)
                  missing   %
snowdepth          101723 100
ice_accretion_3hr  101648 100
skyc4              101634 100
skyl4              101634 100
ice_accretion_6hr  101607 100
ice_accretion_1hr  101192  99
peak_wind_gust      97603  96
peak_wind_drct      97603  96
peak_wind_time      97603  96
skyc3               93539  92
skyl3               93539  92
gust                87970  86
skyc2               76418  75
skyl2               76418  75
wxcodes             71464  70
skyl1               34243  34
mslp                23579  23
p01i                10800  11
drct                 9525   9
feel                 1909   2
sknt                 1836   2
skyc1                 372   0
dwpf                  363   0
relh                  363   0
tmpf                  329   0
vsby                  259   0
alti                  227   0
station                 0   0
valid                   0   0
metar                   0   0
>
```

Selected Variables

- Statio:three or four character site identifier
- Valid:timestamp of the observation
- tmpf:Air Temperature in Fahrenheit, typically @ 2 meters
- Dwpf: Dew Point Temperature in Fahrenheit, typically @ 2 meters
- Relh: Relative Humidity in %
- Drct: Wind Direction in degrees from *true* north
- Sknt:Wind Speed in knots
- P01i: One hour precipitation for the period from the observation time to the time of the previous hourly precipitation reset.
- Alti: Pressure altimeter in inches
- Mslp: Sea Level Pressure in millibar
- Vsby: Visibility in miles
- Skyc1:Sky Level 1 Coverage
- Skyl1:Sky Level 1 Altitude in feet
- Feel: Apparent Temperature (Wind Chill or Heat Index) in Fahrenheit

Cleaning Data

Separate ‘valid’ column into four columns:Year, month, day, hour

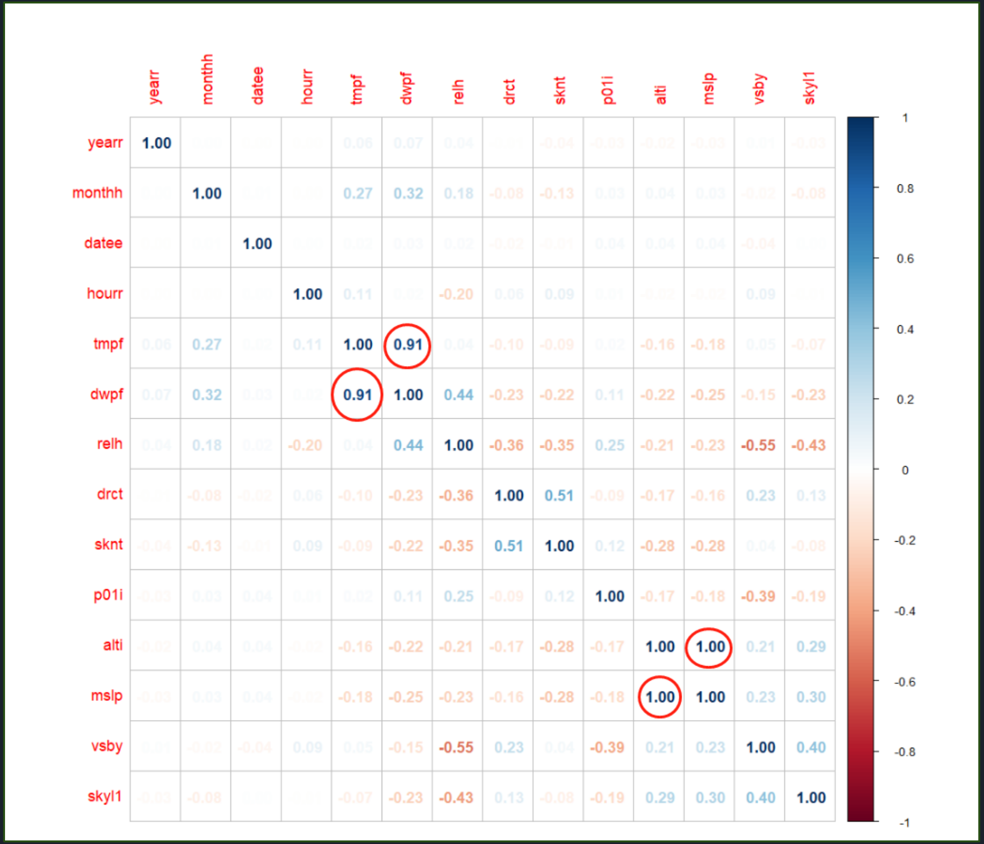
Fill in NaN values: There are still many NaN values in our dataframe, since our variables are about weather, we use the ‘Last Observation Carried Forward’ method to fill those NaN values.

Check Correlation:

dwpf & tempf have high correlation of 0.91

alti & mslp have high correlation of 1

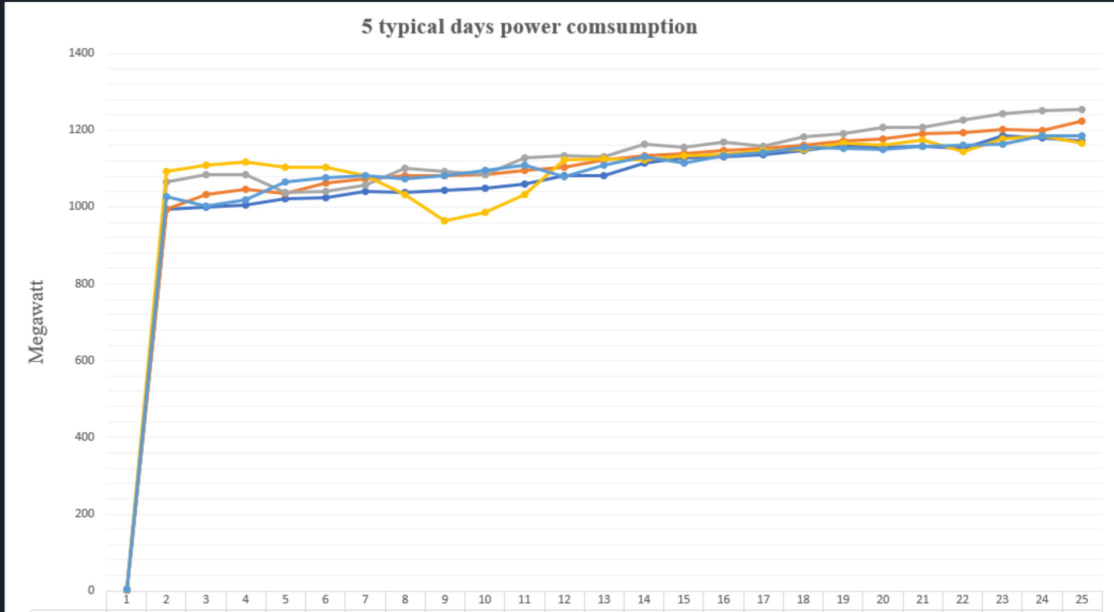
We drop dwpf and alti columns.



Multiple Linear Regression Model

```python
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import cross_val_score
lr = LinearRegression().fit(X_train, y_train)
np.mean(cross_val_score(lr, X_train, y_train, cv=5))
```

```
0.11217934523843542
```



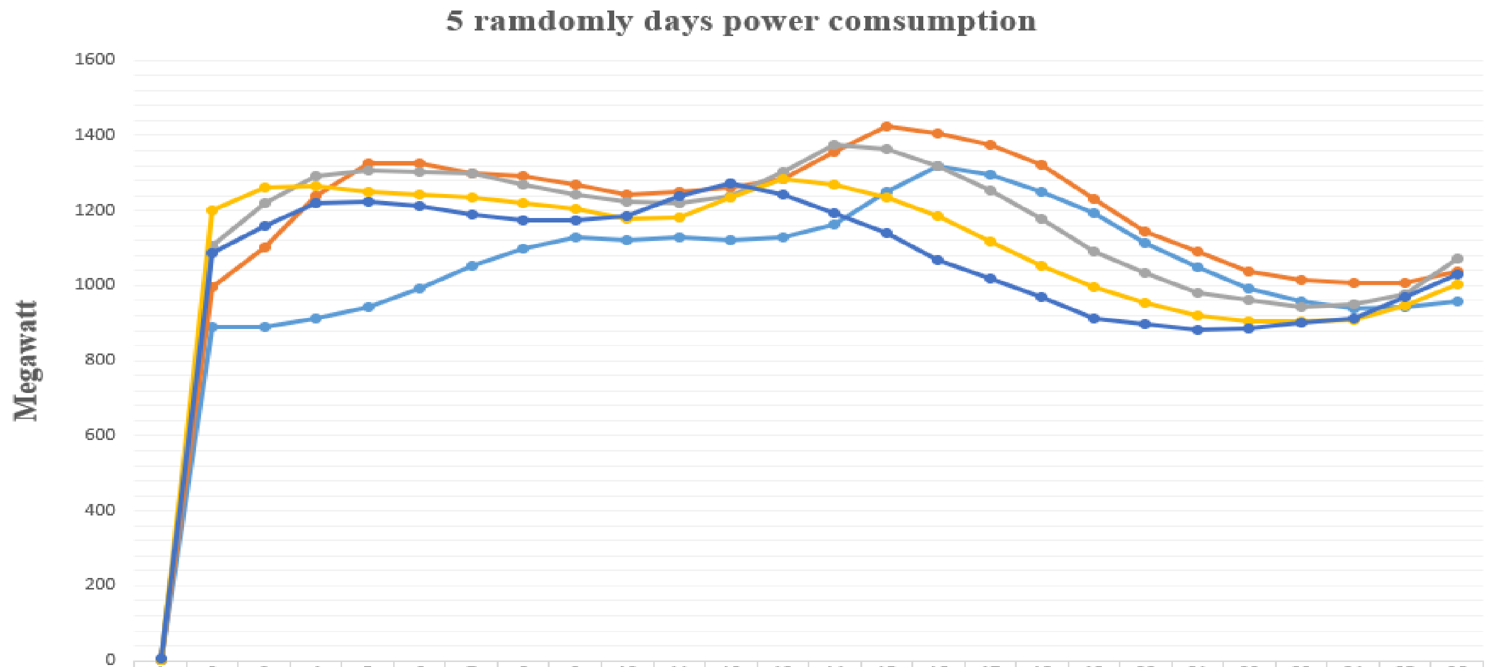
	1	2	3	4	5	6	7	8	9	10	11	12	13
Series1	1	995.2	999.13	1006.4	1021.4	1024.8	1039.9	1036.9	1044.1	1049.2	1060.8	1081.6	1081.7
Series2	2	995.09	1031.4	1047.3	1036.4	1064.2	1073.8	1080.6	1082.9	1086	1095.1	1103.9	1121.8
Series3	3	1065.6	1083.6	1085.5	1036.8	1040.6	1056.9	1099.8	1093	1085.9	1127.2	1133.2	1130.9
Series4	4	1092.2	1108.1	1116.9	1103.4	1104.8	1082.4	1033.7	963.63	986.67	1032.1	1122.3	1126
Series5	5	1027.4	1001.4	1018.7	1066.2	1075.2	1082.3	1073.9	1082.4	1095	1109.6	1078.4	1109.1

	14	15	16	17	18	19	20	21	22	23	24	25
Series1	1114.7	1128.6	1130.3	1136.8	1147.2	1160.1	1155.9	1158.6	1152.8	1185.4	1181.6	1171.8
Series2	1134.9	1140.4	1146.7	1154.2	1161.6	1172.2	1176.6	1190.1	1195.2	1201.9	1200.7	1222.8
Series3	1164.9	1155.1	1169.3	1157.6	1184.1	1192.7	1207.1	1208.6	1227.2	1242.6	1252.7	1253.1
Series4	1122.3	1134.6	1137.6	1148.8	1149.6	1167.5	1162.4	1174.1	1144.1	1178.5	1185.6	1167.9
Series5	1131.6	1113.5	1134.4	1141.2	1155.3	1153.4	1151	1157.3	1162.4	1163.6	1186.6	1184.6

Predict Value & Real value

	Predict Value	Real value
0	1209.360154	1102.033
1	1177.352581	1190.984
2	1075.221780	1177.578
3	1091.009084	1172.951
4	1041.104234	859.464
5	902.133435	1040.533
6	1198.785909	1393.293
7	1025.423534	1266.172
8	1127.307795	1121.585
9	1105.112692	1090.752

Multiple Linear Regression Model



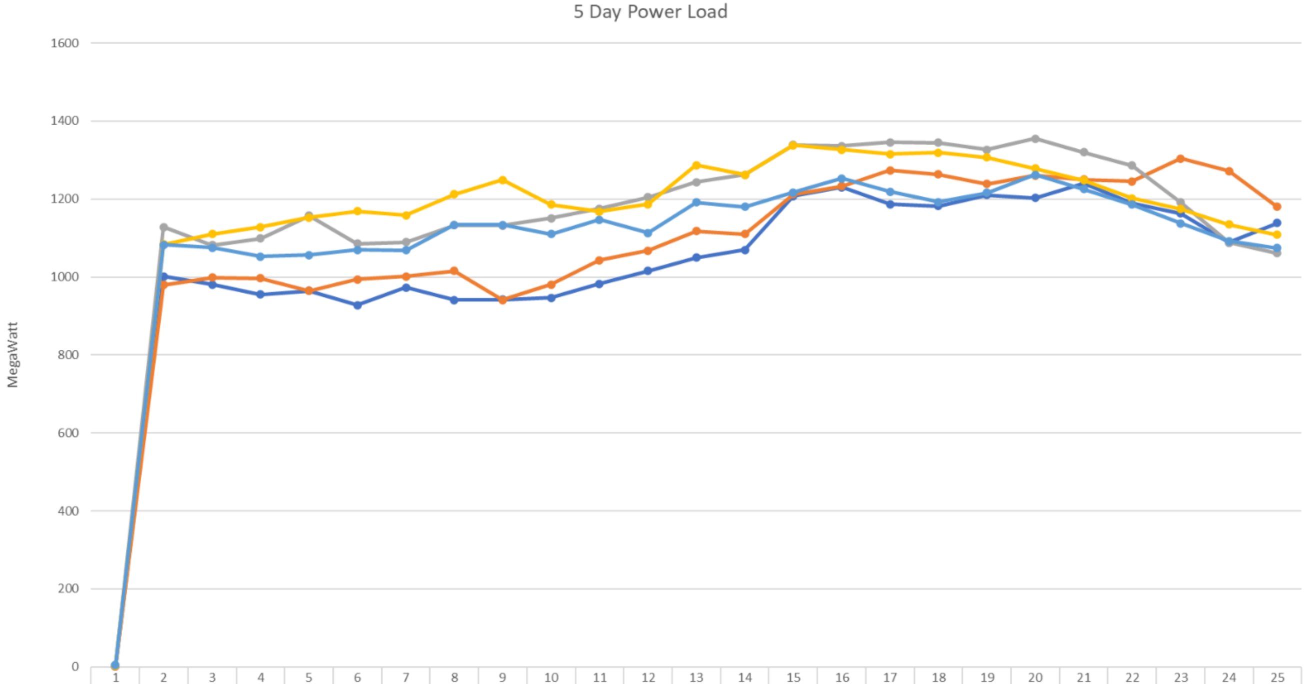
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Series1	1	890.2	888.4	910.7	941.9	991.2	1051.	1097.	1128.	1120.	1130.	1120.	1127.	1161.	1250.	1318.	1293.	1251.	1194.	1112.	1049.	991.2	957.1	938.4	944.1	958.3
Series2	2	995.6	1101.	1238.	1325.	1327.	1297.	1291.	1268.	1243.	1248.	1260.	1284.	1354.	1424.	1406.	1376.	1321.	1230.	1144.	1091.	1039.	1013.	1005.	1005.	1036.
Series3	3	1107.	1221.	1292.	1307.	1301.	1299.	1270.	1243.	1225.	1218.	1239.	1304.	1374.	1364.	1317.	1254.	1178.	1092.	1032.	982.0	960.6	944.6	950.1	978.4	1071.
Series4	4	1200.	1260.	1264.	1251.	1242.	1233.	1219.	1203.	1178.	1183.	1236.	1283.	1270.	1234.	1185.	1118.	1053.	996.8	952.7	918.3	903.3	904.1	909.9	945.9	1003.
Series5	5	1088.	1157.	1218.	1221.	1211.	1189.	1175.	1174.	1185.	1237.	1271.	1243.	1193.	1140.	1069.	1019.	968.3	912.7	896.2	883.8	886.8	902.2	913.8	968.5	1030.

Random Forest Regression Model

```
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor()
np.mean(cross_val_score(rf, X_train, y_train, cv=5))
```

```
0.5031362802852091
```

	Predict Value	Real value
0	1187.84494	1102.033
1	1064.43623	1190.984
2	1051.87970	1177.578
3	1107.31617	1172.951
4	942.14033	859.464
5	1142.30232	1040.533
6	1401.84201	1393.293
7	1007.59489	1266.172
8	1098.34358	1121.585
9	1066.53211	1090.752



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Series1	1	1001.3	980.48	955.363	964.516	928.072	973.101	940.547	941.628	947.113	982.526	1015.34	1049.18	1069.16	1207.33	1229.91	1186.18	1181.92	1210.41	1202.67	1240.62	1188.99	1163.33	1088.74	1138.15
Series2	2	979.655	998.223	996.344	964.757	993.979	1001.24	1015.4	941.429	980.395	1042.63	1067.43	1117.21	1109.78	1212.22	1232.98	1273.66	1263.38	1239.04	1260.62	1250.13	1245.04	1304.16	1271.51	1180.35
Series3	3	1128.36	1082.05	1099.07	1157.16	1085.54	1089.52	1132.93	1132.22	1150.97	1174.95	1204.35	1243.39	1261.77	1338.32	1335.47	1345.71	1344.68	1326.89	1354.6	1319.54	1285.61	1191.13	1088.52	1061.25
Series4	4	1083.71	1110.34	1128.38	1152.36	1168.26	1158.34	1211.71	1248.61	1185.47	1167.9	1186.19	1287.02	1262.17	1337.39	1326.21	1315.42	1318.99	1306.39	1277.84	1248.35	1201.33	1174.59	1134.29	1107.97
Series5	5	1082.32	1075.46	1052.14	1056.01	1069.54	1068.42	1133.85	1133.6	1110.28	1147.11	1112.76	1191.16	1179.63	1217.11	1253.06	1218.3	1192.47	1215.45	1261.95	1225.65	1185.27	1137.75	1092.37	1074.64

Axis Title

Learn from those Models

The variable ‘Hour’ would definitely affect power grid load, so we use it as one of the input variables.

There are patterns of the load curve. For example, in most of the days, 3 am is the time where power grid load meet its minimum. From 3 am to 6 pm, the power grid load continue increasing and meet its maximum at 6 or 7 pm. Then it continues decreasing to meet its minimum at 3 am.

As for our models’ prediction, we did not do well. As shown in previous slides, our multiple linear regression model predict that the power grid load would continue to increase in a given day.

For our random forest regression model, the results are slightly better than the Linear Regression model. The model successfully predict the pattern that from 6 pm to 3 am, the power grid load would continue to decrease to daily minimum, then continues to increase to daily maximum at 6 pm.

If we have more time, we would build a time series model, this should predict better outcomes that fits the power grid load in real data!

Thank You!