With weather

**Radar Graph**

**Graph description:**

There are 16 features in radar graph, included weather factors. Such as: destination, origin, aircraft type, aircraft manufacturer, airline, estimated arrival time(hrs), wind speed, gust speed, etc. Radar graph shows each attribute’s feature importance score.

**File:**

radar\_feature\_w.html

radar\_feature.html

**Code:**

**Without weather:**

﻿#####################################################

#1st graph

#radar gram(without weather)

attr\_list = ['aircraft type',

'origin', 'destination', 'filed departure time(week)',

'filed departure time(hr)', 'estimated arrival time(hr)', 'airline',

'aircraft manufacturer','aircraft type']

feature\_importances=[0.08550837, 0.14906092, 0.1630565, 0.25129193, 0.0653423, 0.16155004,

0.08253172, 0.04165823]

data = [go.Scatterpolar(

r = feature\_importances,

theta = attr\_list,

fill = 'toself'

)]

layout = go.Layout(title = 'Radar Graph for Feature Importance',

polar = dict(

radialaxis = dict(

visible = True,

range = [0, 0.30]

)

),

showlegend = False

)

fig = go.Figure(data=data, layout=layout)

py.plot(fig, filename = 'radar')

plotly.offline.plot(fig, filename = 'radar\_feature.html')

#radar gram(with weather)

attr\_list = ['aircraft type', 'filed airspeed(kts)', 'filed altitude',

'origin', 'destination',

'filed departuretime(hr)', 'estimated arrival time(hr)', 'airline',

'aircraft manufacturer',

'cloud altitude','temperature','dew point','visibility',

'wind speed','gust speed','aircraft type']

feature\_importances=[0.07676359, 0.17079143, 0.10234315, 0.03712042, 0.09973636, 0.05304062,

0.04998065, 0.04114598, 0.04027695, 0.11216917, 0.04421938, 0.06899675,

0.00545371, 0.08449094, 0.0134709 ]

data = [go.Scatterpolar(

r = feature\_importances,

theta = attr\_list,

fill = 'toself'

)]

layout = go.Layout(title = 'Radar Graph for Feature Importance (with weather factor)',

polar = dict(

radialaxis = dict(

visible = True,

range = [0, 0.20]

)

),

showlegend = False

)

fig = go.Figure(data=data, layout=layout)

py.plot(fig, filename = 'radar\_w')

plotly.offline.plot(fig, filename = 'radar\_feature\_w.html')

**with weather:**

﻿#radar gram(with weather)

attr\_list = ['aircraft type', 'filed airspeed(kts)', 'filed altitude',

'origin', 'destination',

'filed departuretime(hr)', 'estimated arrival time(hr)', 'airline',

'aircraft manufacturer',

'cloud altitude','temperature','dew point','visibility',

'wind speed','gust speed','aircraft type']

feature\_importances=[0.07676359, 0.17079143, 0.10234315, 0.03712042, 0.09973636, 0.05304062,

0.04998065, 0.04114598, 0.04027695, 0.11216917, 0.04421938, 0.06899675,

0.00545371, 0.08449094, 0.0134709 ]

data = [go.Scatterpolar(

r = feature\_importances,

theta = attr\_list,

fill = 'toself'

)]

layout = go.Layout(title = 'Radar Graph for Feature Importance (with weather factor)',

polar = dict(

radialaxis = dict(

visible = True,

range = [0, 0.20]

)

),

showlegend = False

)

fig = go.Figure(data=data, layout=layout)

py.plot(fig, filename = 'radar\_w')

plotly.offline.plot(fig, filename = 'radar\_feature\_w.html')

**Histogram Graph**

**Graph description:**

This histogram shows each airline company’s departure and arrival delay. Through inspection of this graph, we could get direct contrast of delayed flight among those airline companies.

**File: hist.html**

**Code:**

﻿#####################################################

#2nd graph

#histogram

x = df.clean.airline

y = df.clean.delay\_dep

z = df.clean.delay\_arr

data = [

go.Histogram(

histfunc = "count",

y = y,

x = x,

name = "Total Number of Flight"

),

go.Histogram(

histfunc = "sum",

y = y,

x = x,

name = "Number of Depature Delayed Flight"

),

go.Histogram(

histfunc = "sum",

y = z,

x = x,

name = "Number of Arrival Delayed Flight"

)

]

layout = go.Layout(

title='Histogram for depature flights',

xaxis=dict(

title='Airline Company'

),

yaxis=dict(

title='Count'

),

bargap=0.2,

bargroupgap=0.1

)

fig1 = go.Figure(data=data, layout=layout)

fig1['layout'].update(title = "some title")

py.plot(data, filename='hist\_depature')

plotly.offline.plot(data, filename = 'hist.html')