Freight Analysis Framework

Baseline Scenario

The Freight Analysis Framework (FAF) integrates data from a variety of sources to create a comprehensive national picture of freight movements among states and major metropolitan areas by all modes of transportation. It provides a national picture of current freight flows to, from, and within the United States, assigns the flows to the transportation network, and projects freight flow patterns into the future.
The FAF flow matrix described in this report is to forecast future freight shipment weights .
The FAF Forecasts must to be driven by the most up to date macroeconomic assumptions on the short- and long-term trends of the United States .

Challenge addressed by the use case

It's unavailable to get forecasted economic dataSet by industry sector and by geographic region to forecast growth rate for long term (from 2020 to 2040) .
To solve this I assumed to start from the end which it mean to get the final results from the available dataSets (in our case the goods in tons from 2012 to 2018) to calculate the expected growth rate in goods tons between each year and the base year (2012) .
To calculate the growth rate for the next years in short term (from 2019 to 2024) from the available dataSets , I assumed to calculate the growth rate for each year and the previous year for it and get the avarage for all of these growth rates (2012 - 2018) and added it to next year such as 2019 and calculate the growth rate for 2019 based on 2012. This process will repeat to 2024 .

My Goal

The goal is to forecast freight shipment weights in 2024 .

The Model

The dataSet from 2012 - 2018 got divided in term of domestic origin to 22 model which each model contains 6 origin out of 132 domestic origin.

The Evaluations

	LinearRegression	DecisionTreeRegressor	GBTRegressor	RandomForestRegressor
Model_1	0.128567	0.856142	0.927008	0.806357
Model_2	0.163305	0.957942	0.965416	0.838849
Model_3	0.151153	0.886418	0.90306	0.810608
Model_4	0.168629	0.94104	0.953556	0.912967
Model_5	0.0825861	0.681653	0.727471	0.559123
Model_6	0.0523374	0.582365	0.479348	0.476045
Model_7	0.080967	0.719116	0.751444	0.564661
Model_8	0.10272	0.862531	0.930598	0.823979
Model_9	0.0857107	0.638907	0.640282	0.684703
Model_10	0.15655	0.937152	0.939739	0.918136
Model_11	0.0711623	0.790661	0.705178	0.776414
Model_12	0.0762228	0.573328	0.610031	0.538463
Model_13	0.0940348	0.947823	0.949527	0.852151
Model_14	0.13747	0.910271	0.912732	0.866331
Model_15	0.105654	0.919193	0.917796	0.892296
Model_16	0.107502	0.862861	0.868508	0.830937
Model_17	0.190484	0.879671	0.782876	0.833701
Model_18	0.120456	0.971409	0.925193	0.914135
Model_19	0.10167	0.409774	0.411038	0.523731
Model_20	0.13434	0.688126	0.689623	0.713678
Model_21	0.102864	0.944469	0.946948	0.884075
Model_22	0.164726	0.946272	0.95345	0.938329

The Deployment

Used RandomForestRegressor algorithm to apply it in spark standalone cluster.