

REDUCING HEALTHCARE COSTS THROUGH PATIENT ENCOUNTER ANALYSIS

MSDS436: Analytics Systems Engineering

Megan Bohan

Ferdynand Hebal

Andrew Lee

EXECUTIVE SUMMARY

Objectives: Reduce costs for a nationwide hospital system by identifying groups of patients who are predicted to have high costs and developing cost reduction programs.

Results: The bulk of the predicted costs come from patients aged 65-90. Patients with chronic conditions have higher median predicted costs than patients who do not have those chronic conditions. Patients with heart disease, diabetes, heart failure, and kidney disease are predicted to have a high sum of costs when taking patient volumes into consideration.

Conclusion: The hospital system should develop preventative health and monitoring programs for patients with risk factors for or already diagnosed with heart disease, diabetes, heart failure, and kidney disease. These programs should be prioritized for patients between ages 65-90, and states with high volumes of patients.

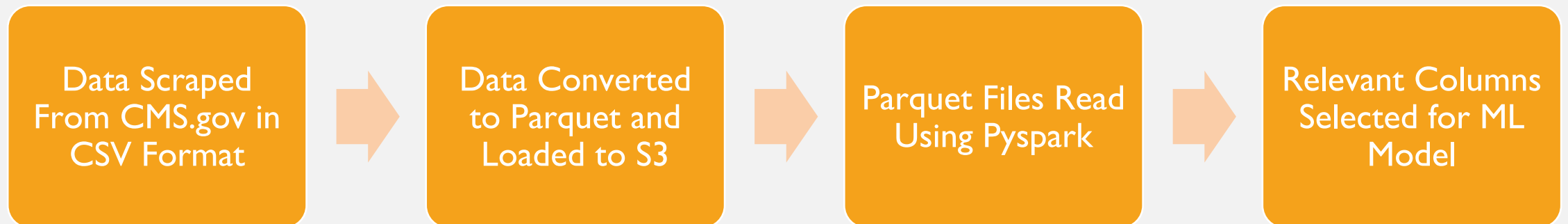
RESEARCH OBJECTIVES

- A nation wide hospital system is faced with increasing costs and is looking for help to decrease costs without negatively impacting patient care. **They are looking to identify groups of patients who are at risk of being high cost and develop programs that will reduce the overall cost.**
- Research has shown that patients with chronic conditions are high cost, and that the top 5% of high cost patients contribute to 55% of healthcare costs each year (van der Wees et al., 2018). Research has also shown that socio-economic factors contribute to overall health outcomes (Magnan, 2019).
- The Centers for Medicare & Medicaid's Synthetic Public Use data set contains both diagnosis and demographic data related to patient encounters. This data will be used to create a machine learning model that will predict the risk of a patient being high cost.
- The cost prediction information can be used to identify patient populations to proactively engage in health programs to reduce their overall cost to the healthcare system.

DATA SETS AND DATA PREPARATION

CMS Synthetic Public Use File (CMS.gov, 2019):

- Beneficiary Summary
- Inpatient Claims
- **Outpatient Claims
- **Prescription Drug Events

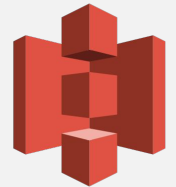


** Data was scraped and formatted but not used in analysis

DATA INGESTION, ANALYSIS, PREPARATION



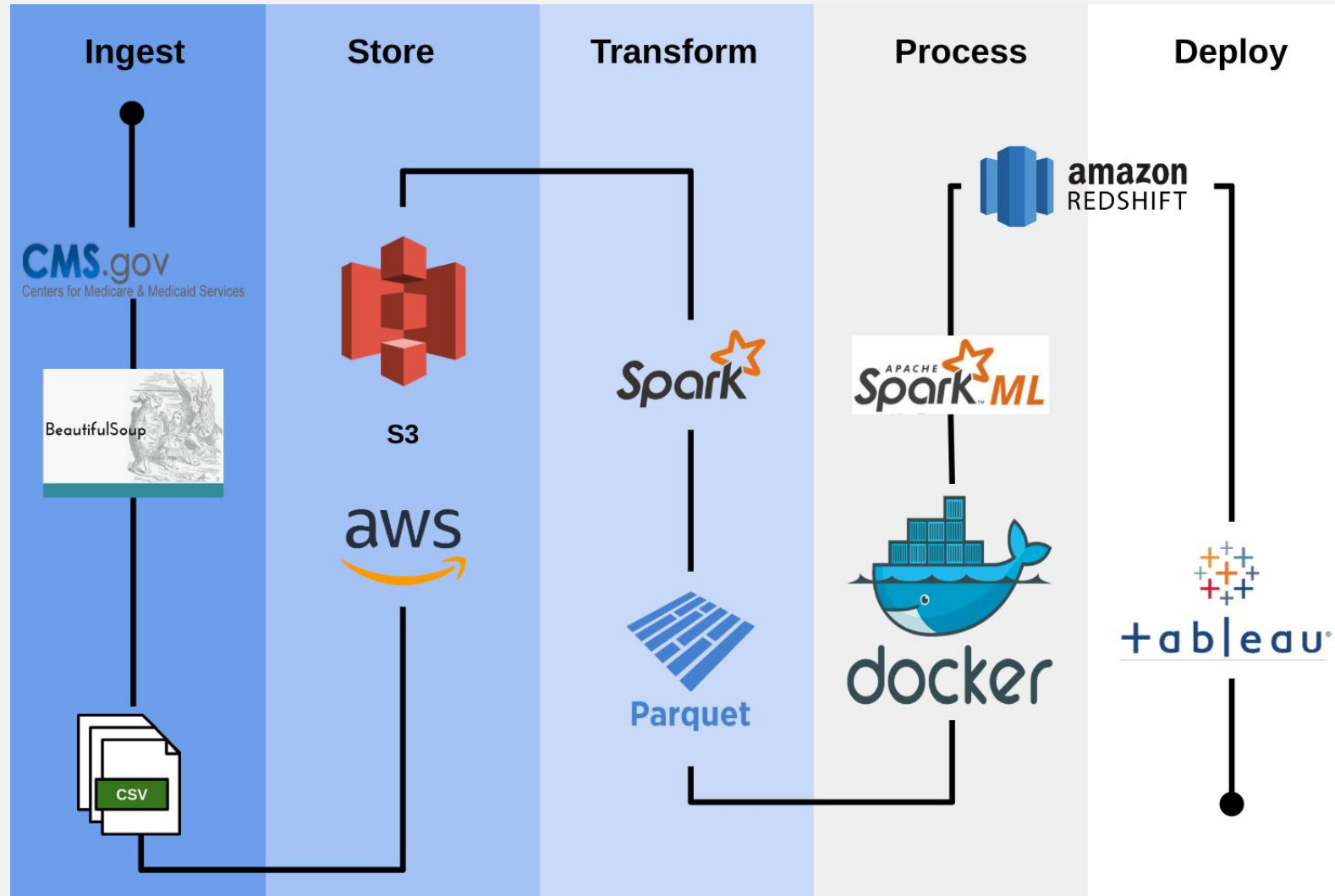
- CMS data was scraped using Python and BeautifulSoup
- All files including 5 table schemas and 135,666,373 patient records ingested
- Amazon S3 was the first stopping point for ingested data
- CSV files converted to Parquet files to compress size and speed up read
- Webscraping, data ingestion, and transformation script was containerized using Docker and deployed in Amazon EC2 instance
- PySpark jobs aggregated data and applied transformations
- ML script was built in Jupyter and reduced to a scalable Python script
- ML script was containerized using Docker and deployed in Amazon EC2 instance
- Feature engineering, model training and evaluation conducted in container scripts
- Analysis results output to Parquet file in Amazon Redshift



METHODOLOGY AND TOOLS USED

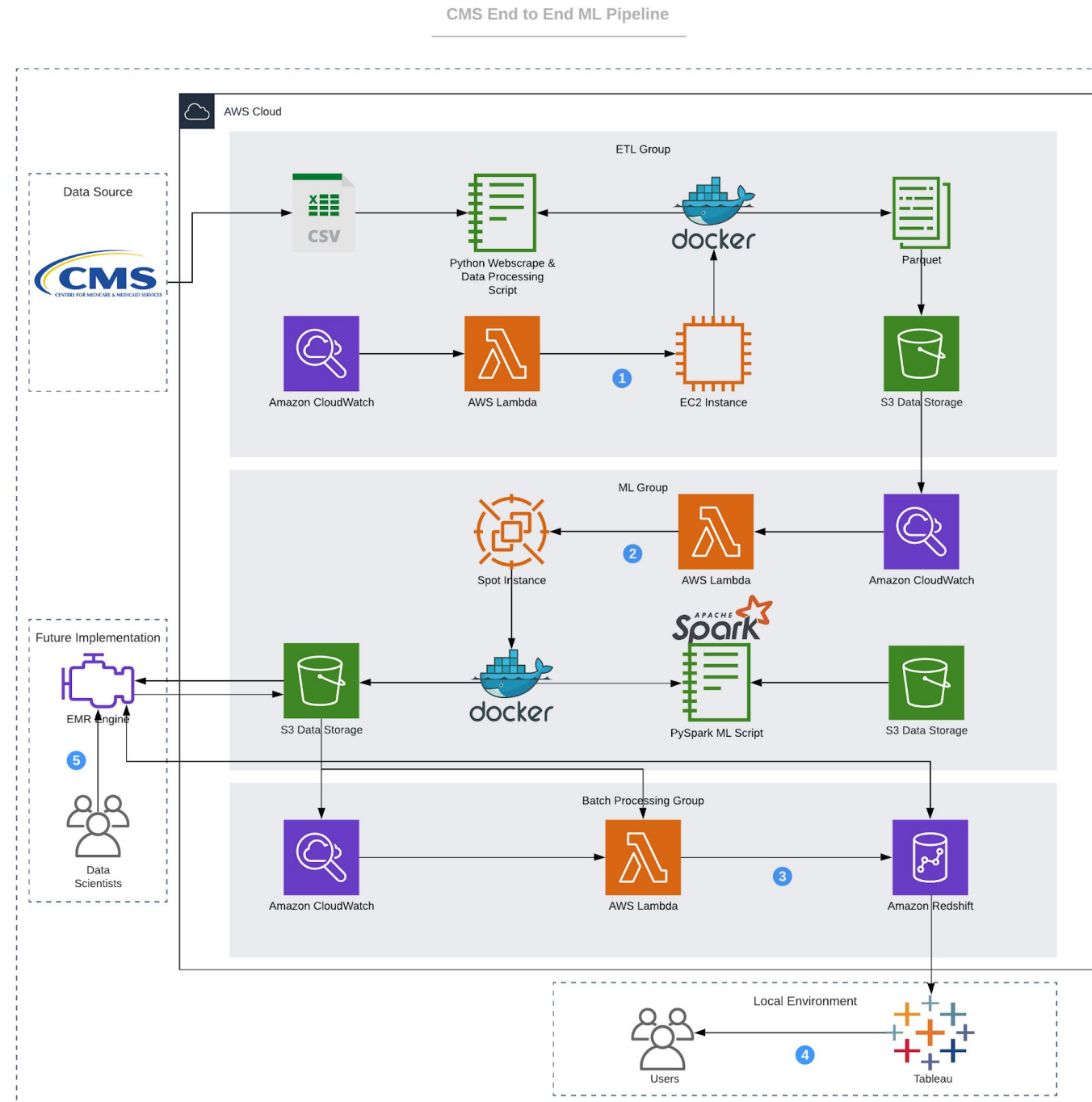
- Python & BeautifulSoup - Web Scraping
- PySpark - Data Ingestion and Transformation
- Boto3 - Interface with Amazon APIs
- AWS API
- Amazon CLI
- AWS Lambda - Automation/Scheduling Scripts and Batch Processing
- Paramiko - SSHv2 Protocol to Connect to EC2, Open Shell, and Run Docker Image
- AWS S3 - Storage
- Spark ML - Linear Regression
- Docker - Containerization of ETL Process, Feature Engineering, Model Training, and Deploy
- Node.js - Load Data into Redshift
- Redshift - Data Warehouse and Insight Result Storage for Deployment to Tableau
- Tableau - Data Visualization and Insight Reporting

SYSTEM ARCHITECTURE



SYSTEM ARCHITECTURE

- End to end pipeline is fully automated with a near serverless architecture.
- Automation is implemented with a series of Amazon Lambda functions that starts with an initial function that is scheduled on regular intervals within Amazon CloudWatch.
- Each script is containerized within Docker images where each VM instance is started and stopped through triggers for cost efficiency.



System Elements

1 Using Amazon CloudWatch, a scheduled event triggers a Lambda function that starts an Amazon EC2 instance that contains a Docker image. Once the instance is started another function is triggered to connect to the instance via SSH through a virtual environment setup in Lambda. A shell is started that runs a Python script within the Docker image that scrapes CSV files from the CMS website, transforms and converts the data into Parquet files, and uploads them to Amazon S3. After processing, an event is logged triggering another Lambda function that stops the instance.

2 The event that is logged after processing triggers another Lambda function that starts an EC2 Spot Instance. Once the instance is started a Lambda function is triggered that connects to the Spot Instance via SSH through a virtual environment, and starts a shell that runs a machine learning script within a Docker image. Using PySpark, data is read from the files in S3 and a linear regression model is trained. Once trained, predictions are made and processed into a Parquet file that is uploaded into S3. The processing is logged and a Lambda function is triggered to set the Spot Instance to hibernate.

3 Through CloudWatch logs, the processed event triggers multiple Lambda functions that contains files and scripts written in Node.js that loads the Parquet files from the folders in S3 to Amazon Redshift. Each Lambda function that loads Parquet files from an S3 folder into their respective table is set to trigger sequentially. That is, once a folder is loaded, an event is sent to CloudWatch logs, and the next Lambda function is triggered, which loads the next folder.

4 Once files are processed into Redshift, updated data is available for manipulation, analysis, and visualization either through direct queries in the Redshift query editor or a BI and visualization tool such as Tableau. Users are able to directly access data from Redshift when connected to the cluster using software like Tableau.

5 It is recommended that a future iteration implements an EMR cluster(s) for direct access by Data Scientists for more robust modeling and predictive analytics. The EMR cluster(s) can directly access S3 or Redshift and can be used to transform and move data in and out of storage or databases/datawarehouse.

SYSTEM ARCHITECTURE AUTOMATION



Amazon Cloudwatch

aws

Services ▾

Resource Groups ▾

📌

CloudWatch

Dashboards

Alarms

ALARM

INSUFFICIENT

OK

Billing

Logs

Log groups

Insights

Metrics

Events

Rules

Event Buses

ServiceLens

Service Map

Traces

Synthetics

Canaries

Contributor Insights

Settings

Rules > lambda-start-cmss3-ec2

Actions ▾

Summary

ARN ⓘ `arn:aws:events:us-east-1:799687528413:rule/lambda-start-cmss3-ec2`

Schedule

Cron expression `0 23 ? * MON *`

Next 10 Trigger

Date(s)

1. Mon, 16 Mar 2020 23:00:00 GMT

2. Mon, 23 Mar 2020 23:00:00 GMT

3. Mon, 30 Mar 2020 23:00:00 GMT

4. Mon, 06 Apr 2020 23:00:00 GMT

5. Mon, 13 Apr 2020 23:00:00 GMT

6. Mon, 20 Apr 2020 23:00:00 GMT

7. Mon, 27 Apr 2020 23:00:00 GMT

8. Mon, 04 May 2020 23:00:00 GMT

9. Mon, 11 May 2020 23:00:00 GMT

10. Mon, 18 May 2020 23:00:00 GMT

Status **Enabled**

Description Start ec2 instance once a week for data processing from CMS

Monitoring [Show metrics for the rule](#)

Targets

Filter:

« < Viewing 1 to 1 of 1 Targets > »

Type	Name	Input	Role	Additional parameters
Lambda function	start-cmss3-ec2	Matched event		

Using Amazon CloudWatch, an event is set to trigger a Lambda function that is scheduled every week as a cron job.

The initial function starts an EC2 instance. Once the EC2 instance has started, the logged event will trigger another Lambda function that will connect to the instance via SSH, open a shell, and run a script within a Docker image. The script scrapes data files from the CMS website, transforms and converts the data into Parquet files, and loads them into S3. Once the files are loaded, another logged event triggers a Lambda function to stop the EC2 instance.

CloudWatch Logs

[/aws/lambda/start-cmss3-ec2](#)

`arn:aws:logs:us-east-1:799687528413:log-group:/aws/lambda/start-cmss3-ec2:*`

Filter name: **lambda-process-cmss3** Filter pattern: **END RequestId**

☒ Enabled

Delete

SYSTEM ARCHITECTURE AUTOMATION



Amazon CloudWatch

aws

Services

Resource Groups

CloudWatch

Dashboards

Alarms

ALARM

INSUFFICIENT

OK

Billing

Logs

Log groups

Insights

Metrics

Events

Rules

Event Buses

ServiceLens

Service Map

Traces

Synthetics

Canaries

Contributor Insights

Settings

Favorites

Add a dashboard

/aws/lambda/end_cmss3-ec2

/aws/lambda/end-sparkml-ec2

/aws/lambda/process-cmss3

/aws/lambda/redshift-load-beneficiary-summary

(+8)

2020-02-01 (00:00:00) - 2020-03-11 (01:38:32)

fields @log, @timestamp, @message

| sort @timestamp desc

Run query

Actions

Sample queries

Have feedback? Email us.

Logs

Visualization

467 records matched | 467 records (50.8 kB) scanned in 9.4s @ 49 records/s (5.4 kB/s)

#	@log	@timestamp	@message
1	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-11T00:30:40.859-07:00	END RequestId: c0931a68-9c96-4870-a062-2d70ba627f58
2	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-11T00:30:40.859-07:00	REPORT RequestId: c0931a68-9c96-4870-a062-2d70ba627f58 Duration: 2290.32 ms Billed Dura...
3	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-11T00:30:40.838-07:00	4 Stopped Instance ID ['i-0eb9a408ec2b491c8']
4	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-11T00:30:38.568-07:00	START RequestId: c0931a68-9c96-4870-a062-2d70ba627f58 Version: \$LATEST
5	799687528413:cms-annual-cost-predictions	2020-03-11T00:30:37.352-07:00	3 Predictions processed in 0:31:09.332210
6	799687528413:/aws/lambda/sparkml-cms	2020-03-11T00:04:29.661-07:00	REPORT RequestId: 71a76e7a-7786-444a-bcc6-167187b156ca Duration: 423941.84 ms Billed Du...
7	799687528413:/aws/lambda/sparkml-cms	2020-03-11T00:04:29.642-07:00	END RequestId: 71a76e7a-7786-444a-bcc6-167187b156ca
8	799687528413:/aws/lambda/sparkml-cms	2020-03-10T23:59:19.501-07:00	Executing sudo service docker start &&
9	799687528413:/aws/lambda/sparkml-cms	2020-03-10T23:59:19.501-07:00	2 sudo docker run -it ff572cbb6282 python3 /model/pyspark_model.py /bin/bash
10	799687528413:/aws/lambda/sparkml-cms	2020-03-10T23:59:19.452-07:00	Connected to 18.234.139.184
11	799687528413:/aws/lambda/sparkml-cms	2020-03-10T23:58:58.222-07:00	Downloading key
12	799687528413:/aws/lambda/sparkml-cms	2020-03-10T23:57:27.802-07:00	1 Connecting to ec2 instance
13	799687528413:/aws/lambda/sparkml-cms	2020-03-10T23:57:25.700-07:00	START RequestId: 71a76e7a-7786-444a-bcc6-167187b156ca Version: \$LATEST
14	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-10T23:47:23.478-07:00	END RequestId: 7892f25e-513b-4bcf-8753-2b19eaf6cef8
15	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-10T23:47:23.478-07:00	REPORT RequestId: 7892f25e-513b-4bcf-8753-2b19eaf6cef8 Duration: 2117.74 ms Billed Dura...
16	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-10T23:47:23.438-07:00	Stopped Instance ID ['i-0eb9a408ec2b491c8']
17	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-10T23:47:21.341-07:00	START RequestId: 7892f25e-513b-4bcf-8753-2b19eaf6cef8 Version: \$LATEST
18	799687528413:cms-annual-cost-predictions	2020-03-10T23:47:19.258-07:00	Predictions processed in 0:31:00.285525
19	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-10T21:44:49.685-07:00	END RequestId: 1ea04def-4591-4a69-a3b7-cdbfb24d0590
20	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-10T21:44:49.685-07:00	REPORT RequestId: 1ea04def-4591-4a69-a3b7-cdbfb24d0590 Duration: 2069.56 ms Billed Dura...
21	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-10T21:44:49.645-07:00	Stopped Instance ID ['i-0eb9a408ec2b491c8']
22	799687528413:/aws/lambda/end-sparkml-ec2	2020-03-10T21:44:47.596-07:00	START RequestId: 1ea04def-4591-4a69-a3b7-cdbfb24d0590 Version: \$LATEST

1. After the files have been processed into S3, a Lambda function starts an EC2 instance, which triggers another Lambda function to connect to the instance via SSH.

2. Once connected, a shell is started that runs a script within a Docker image that trains a linear regression model, and processes predictions.

3. Once the predictions are processed, they are uploaded to S3 as a Parquet file.

4. Another Lambda function is then triggered from the logs and the EC2 instance is stopped.

SYSTEM ARCHITECTURE AUTOMATION



Amazon Cloudwatch

aws

Services

Resource Groups

CloudWatch

Dashboards

Alarms

ALARM

INSUFFICIENT

OK

Billing

Logs

Log groups

Insights

Metrics

Events

Rules

Event Buses

ServiceLens

Service Map

Traces

Synthetics

Canaries

Contributor Insights

Settings

Favorites

Add a dashboard

/aws/lambda/redshift-load-beneficiary-summary

15m 30m 1h 6h 12h 1d custom

fields @log, @timestamp, @message
| sort @timestamp desc
| limit 20

Run query

Actions

Sample queries

Have feedback? Email us.

Logs

Visualization

5 records matched | 5 records (910 B) scanned in 2.7s @ 1 records/s (341.335 B/s)

#	@log	@timestamp	@message
1	799687528413:/aws/lambda/redshift-load-beneficiary-summary	2020-03-10T16:34:54.201-07:00	END RequestId: c3484b13-7177-4f5a-bd38-a2a3197cce48
2	799687528413:/aws/lambda/redshift-load-beneficiary-summary	2020-03-10T16:34:54.201-07:00	REPORT RequestId: c3484b13-7177-4f5a-bd38-a2a3197cce48 Duration: 36036.16 ms Billed Dur...
3	799687528413:/aws/lambda/redshift-load-beneficiary-summary	2020-03-10T16:34:18.380-07:00	2020-03-10T23:34:18.366Z c3484b13-7177-4f5a-bd38-a2a3197cce48 INFO Executing set search...
4	799687528413:/aws/lambda/redshift-load-beneficiary-summary	2020-03-10T16:34:18.180-07:00	2020-03-10T23:34:18.143Z c3484b13-7177-4f5a-bd38-a2a3197cce48 INFO Connecting to redshi...
5	799687528413:/aws/lambda/redshift-load-beneficiary-summary	2020-03-10T16:34:18.135-07:00	START RequestId: c3484b13-7177-4f5a-bd38-a2a3197cce48 Version: \$LATEST

- Once predictions are processed and loaded into S3, a Lambda function is triggered from the logged event that starts a process to ingest updated files into Redshift from S3 folders. The Redshift loader is separated into 5 Lambda functions, each set to trigger after one folder is loaded.
- The function first connects to the Redshift database with the user name and password.
- The function then runs a SQL statement that copies the data from the Parquet files in a folder into Redshift.
- Once completed, the last logged message triggers another Lambda function that loads the next folder into Redshift.

MACHINE LEARNING MODEL DEPLOYING ML MODELS TO PRODUCTION

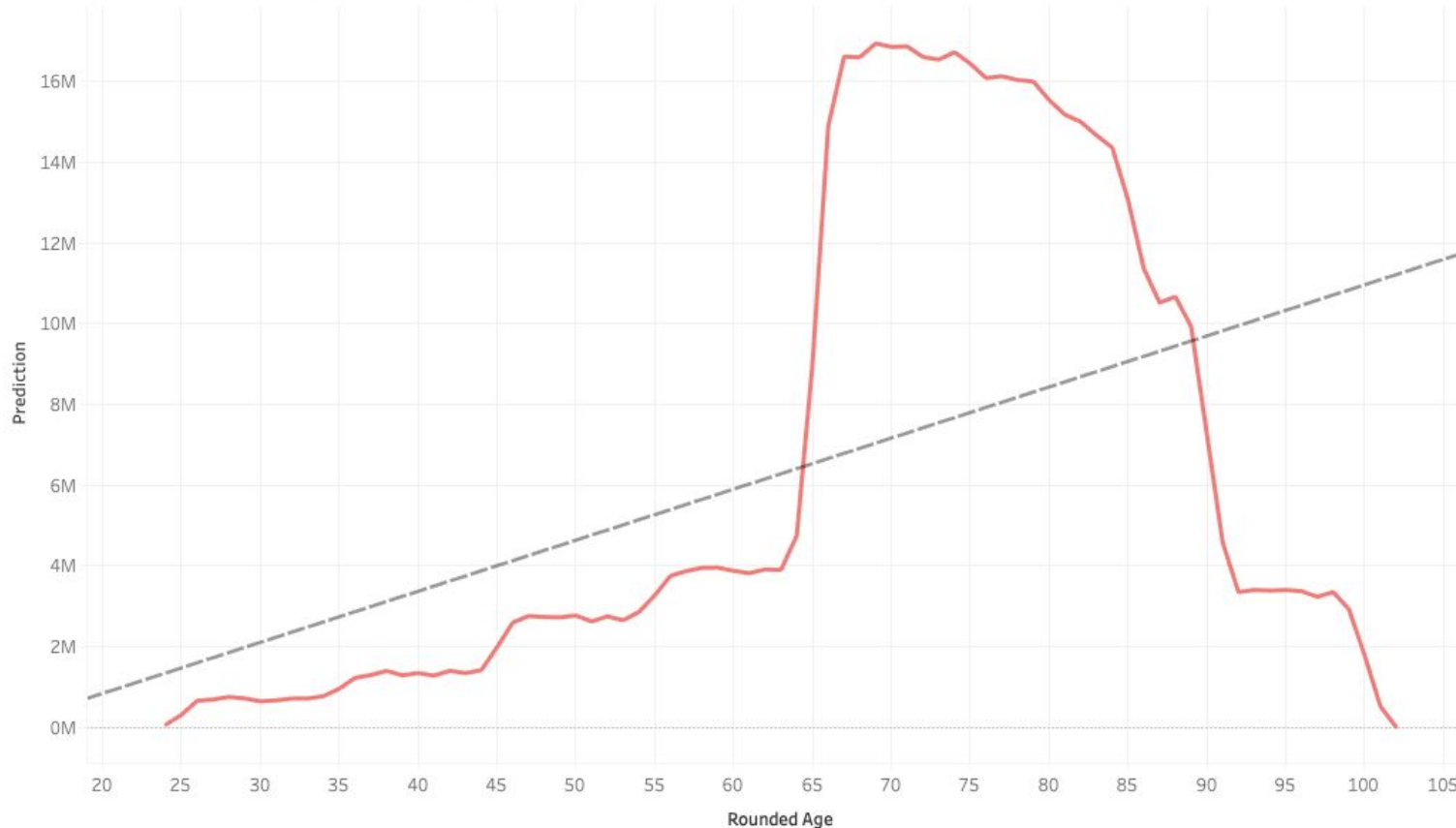


- Parquet files read to spark dataframes and processed with PySpark pipeline functions to transform to appropriate format and data types for ML
- Transformed data random split 0.70. and 0.30 to train and test data sets
- Linear Regression model generated using patient's age, gender, race, and state with the encounter diagnosis code, procedure code, and common procedure code to predict annual cost.
- Predictions merged with patient chronic condition data and output into a Parquet file for evaluation.
- Machine learning script was containerized with Docker and image pushed to Dockerhub.



REPORTING

Sum of Predicted Cost by Patient Age

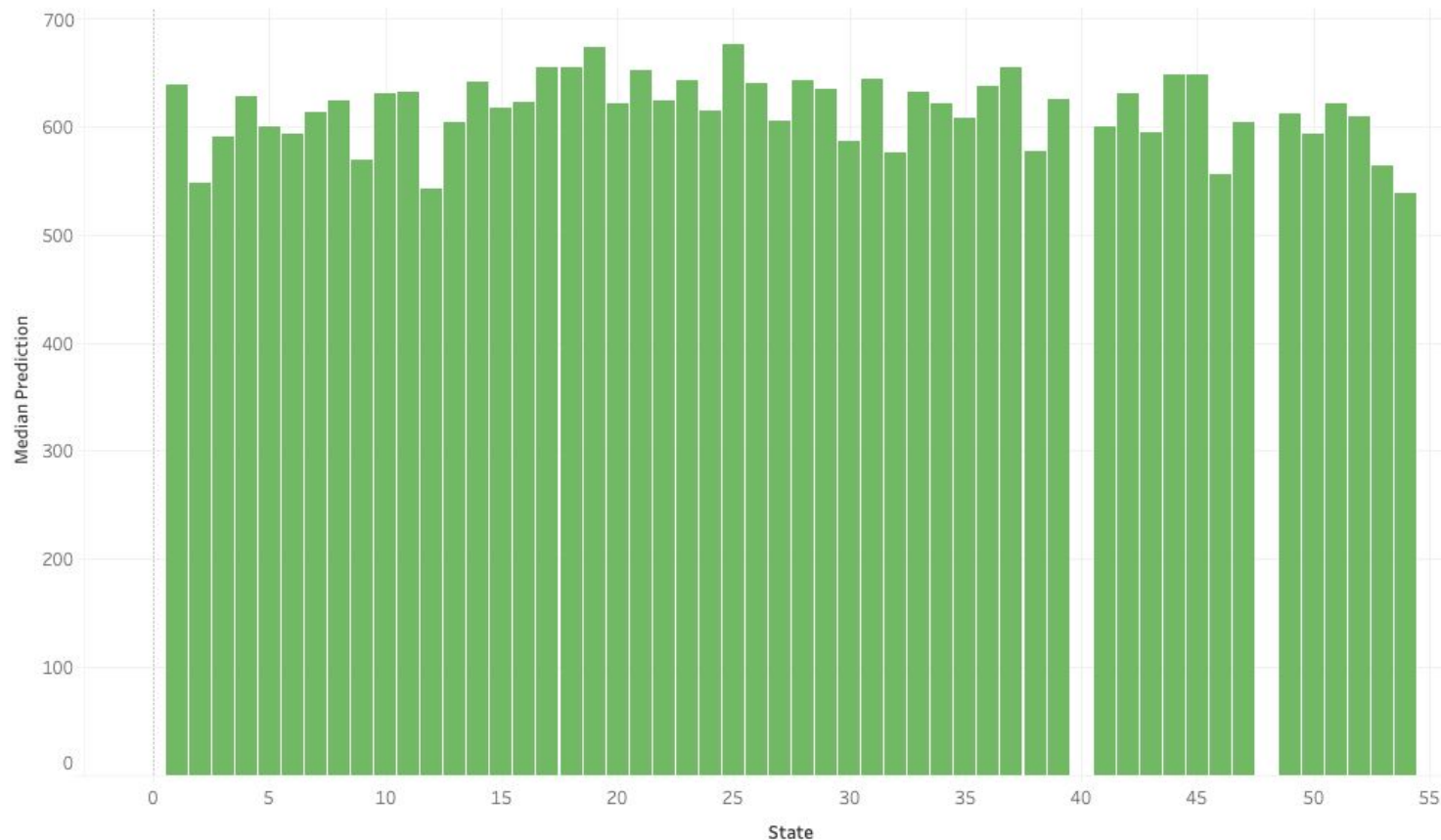


The sum of the predicted costs increases with age, with the bulk of the predicted costs are coming from patients aged 65-90.



REPORTING

Median Cost by State



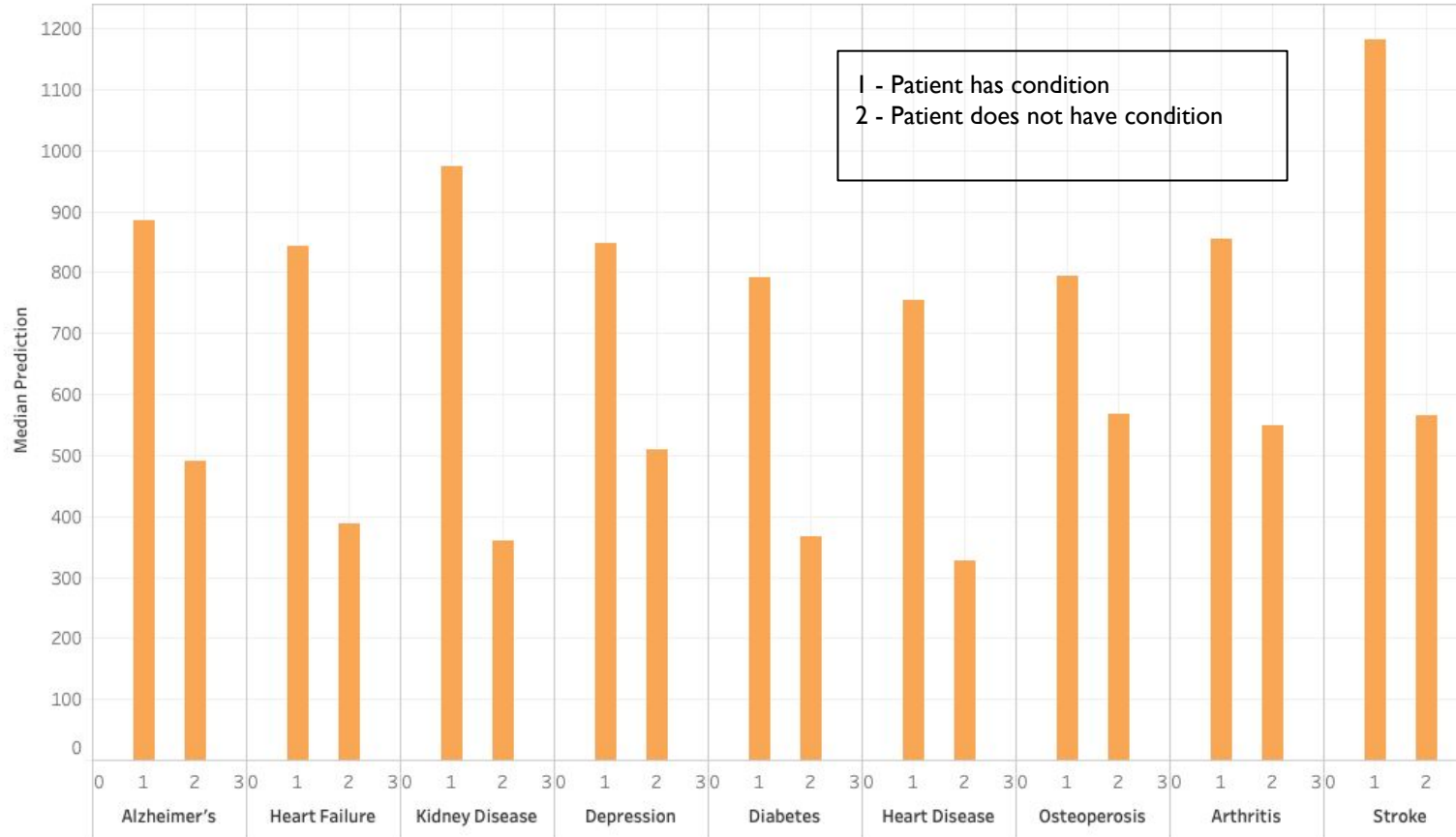
There is not a large difference in median predicted cost by state. This suggests that states with higher volumes of patients will incur higher costs.

* States have been given an ID number to protect patient confidentiality.



REPORTING

Median Cost by Chronic Condition

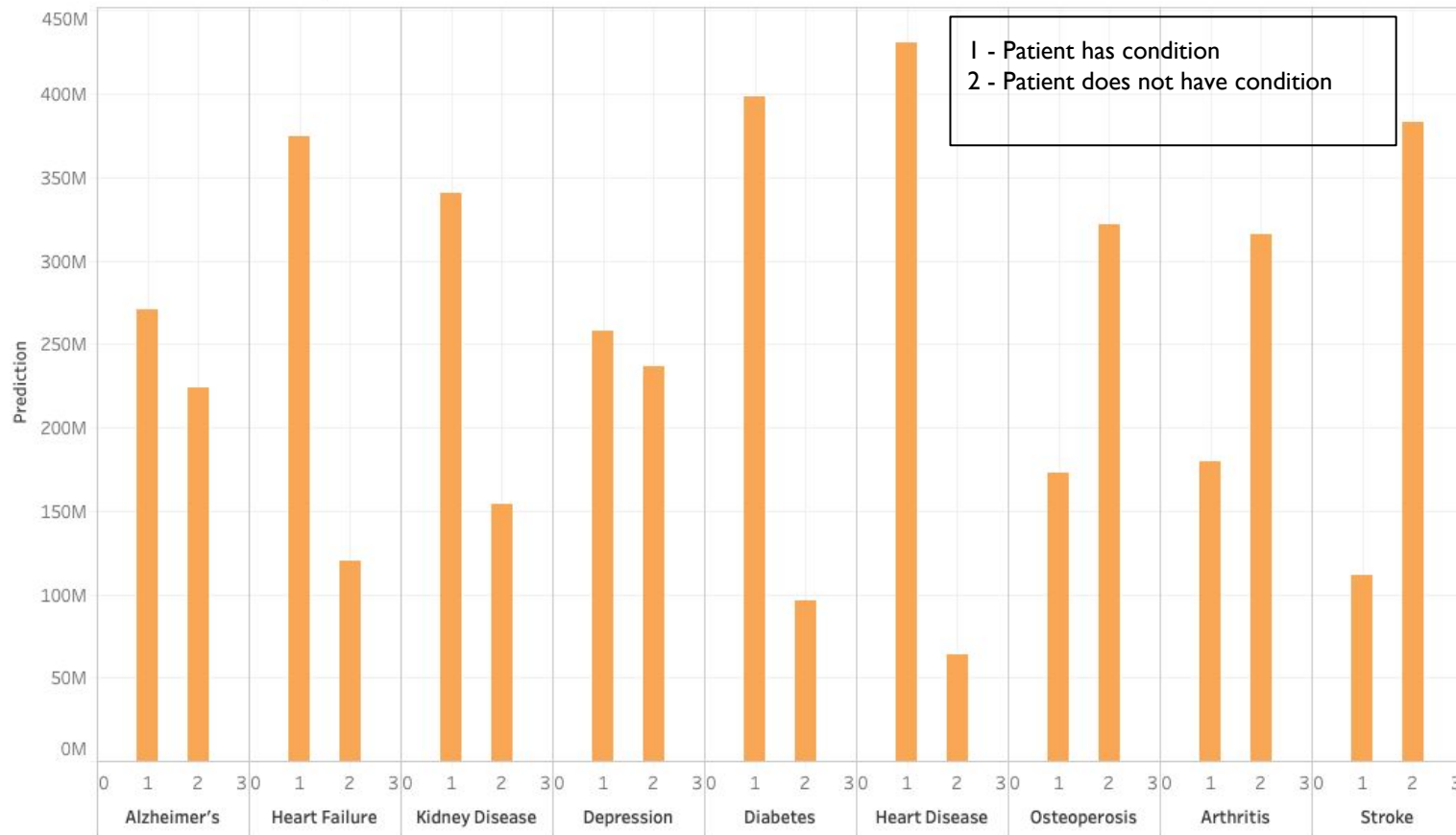


There is an increase in costs for patients who have chronic conditions vs patients who do not.



REPORTING

Sum of Predicted Cost by Chronic Condition



The sum of predicted costs by chronic condition takes patient volumes into account.

This view shows that patients with heart disease, diabetes, heart failure, and kidney disease are predicted to have high overall costs.



RECOMMENDATIONS

- The hospital system should develop preventative health programs for patients with risk factors for heart disease, diabetes, heart failure, and kidney disease.
- The hospital system should monitor patients already diagnosed with these conditions closely to ensure they are following treatment plans.
- The hospital system should prioritize preventative health programs and patient monitoring for patients between 65 and 90, and in states with high patient volumes.

LESSONS LEARNED

Technical

- Data ingestion, processing, transformation, and machine learning could have been accomplished in a completely serverless process.
- Automation process would have been better implemented and organized using AWS Step Functions.
- Amazon EMR was not utilized, and would have been a valuable feature to add.
- Utilizing Amazon ECS for Docker images would have been a great addition to the system architecture.
- Staging tables for Amazon Redshift would have been the ideal setup for data loading rather than copy and deletion.
- Instead of Lambda Functions for batch processing to Amazon Redshift, Amazon Kinesis Firehose would have been a better direction to reliably load data while adding stream processing to the architecture for efficiency.

LESSONS LEARNED

Technical (Continued)

- Rather than waterfall, an iterative methodology should have been utilized for better development.
- Parquet files are significantly smaller than CSV files (12 GB vs 3 GB for the CMS data) and are an efficient and cost effective way of processing data for analysis and transformation.

Analysis

- Test and evaluate other machine learning modeling methods. This was not accomplished due to limited time frame and focus was on building out a working Minimum Viable Product (MVP).
- Data on prescription drug events and outpatient claims were not included in the machine learning model. Though not necessary, they may have been useful in an additional model.

LESSONS LEARNED

Analysis (Continued)

- Predictions on annual cost were in line with the leading health problems affecting the total population in America today.

Team Process

- Weekly meetings and task overview was paramount in successfully deploying an MVP.
- Active peer review made a significant contribution in building out a functional system with fruitful insights and analysis.
- Separation of responsibilities and tasks helped to efficiently and effectively complete the project.
- Utilizing different skill sets and experiences ensured optimal efficiency and project development.

REFERENCES

- CMS 2008-2010 DATA ENTREPRENEURS' Synthetic public use FILE (DE-SYNPUF). (2019, November 22). Retrieved March 07, 2020, from https://www.cms.gov/Research-Statistics-Data-and-Systems/Downloadable-Public-Use-Files/SynPUFs/DE_Syn_PUF
- Magnan, S. (2019, June 18). Social determinants of Health 101 for health CARE: Five plus five. Retrieved March 07, 2020, from <https://nam.edu/social-determinants-of-health-101-for-health-care-five-plus-five/>
- Van der Wees, P., Tanke, M. A., Westert, G. P., Jeurissen, P. P., & Wammes, J. (2018, September 8). Systematic review of high-cost patients' characteristics and healthcare utilisation. Retrieved March 07, 2020, from <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6129088/>

APPENDIX FILES

Processing and Machine Learning Files

- Data Ingest and Processing:
cmss3.py
- Machine Learning:
pyspark_model.py
- Docker:
Dockerfile

Files for Amazon Lambda Functions

- Start and Stop EC2 Instances:
start-cmss3-ec2.zip start-sparkml-ec2.zip
end-cmss3-ec2.zip end-sparkml-ec2.zip
- SSH Connect and Run Docker Image:
lambda-process-cmss3.zip
lambda-sparkml.zip
- Redshift Loader:
redshift-import.zip
lambda-redshift-loader-commands.txt

APPENDIX

DATA SCHEMA

Annual Cost Predictions

Filename: synpuf_ml_output.parquet

Format: Parquet

Size: 660,075 records, 36.2 MB

```
root
|-- ANNUAL_COST: double (nullable = true)
|-- PATIENT_ID: string (nullable = true)
|-- AGE_YRS: double (nullable = true)
|-- GENDER: short (nullable = true)
|-- RACE: short (nullable = true)
|-- STATE: short (nullable = true)
|-- DX: integer (nullable = true)
|-- PX: integer (nullable = true)
|-- HCPCS: integer (nullable = true)
|-- SP_ALZHDMTA: short (nullable = true)
|-- SP_CHF: short (nullable = true)
|-- SP_CHRNKIDN: short (nullable = true)
|-- SP_CNCR: short (nullable = true)
|-- SP_COPD: short (nullable = true)
|-- SP_DEPRESSN: short (nullable = true)
|-- SP_DIABETES: short (nullable = true)
|-- SP_ISCHMCHT: short (nullable = true)
|-- SP_OSTEOPRS: short (nullable = true)
|-- SP_RA_OA: short (nullable = true)
|-- SP_STRKETIA: short (nullable = true)
|-- GENDER_index: double (nullable = true)
|-- RACE_index: double (nullable = true)
|-- STATE_index: double (nullable = true)
|-- SP_ALZHDMTA_index: double (nullable = true)
|-- SP_CHF_index: double (nullable = true)
|-- SP_CHRNKIDN_index: double (nullable = true)
|-- SP_CNCR_index: double (nullable = true)
|-- SP_COPD_index: double (nullable = true)
|-- SP_DEPRESSN_index: double (nullable = true)
|-- SP_DIABETES_index: double (nullable = true)
|-- SP_ISCHMCHT_index: double (nullable = true)
|-- SP_OSTEOPRS_index: double (nullable = true)
|-- SP_RA_OA_index: double (nullable = true)
|-- SP_STRKETIA_index: double (nullable = true)
|-- features: string (nullable = true)
|-- prediction: double (nullable = true)
```

APPENDIX

DATA SCHEMA

Beneficiary Summary

Filename: Multiple Files

Format: Parquet

Total Size: 6,760,520 records, 188.9 MB

```
root
|-- DESYNPUF_ID: string (nullable = true)
|-- BENE_BIRTH_DT: date (nullable = true)
|-- BENE_DEATH_DT: date (nullable = true)
|-- BENE_SEX_IDENT_CD: short (nullable = true)
|-- BENE_RACE_CD: short (nullable = true)
|-- BENE_ESRD_IND: string (nullable = true)
|-- SP_STATE_CODE: short (nullable = true)
|-- BENE_COUNTY_CD: short (nullable = true)
|-- BENE_HI_CVRAGE_TOT_MONS: short (nullable = true)
|-- BENE_SMI_CVRAGE_TOT_MONS: short (nullable = true)
|-- BENE_HMO_CVRAGE_TOT_MONS: short (nullable = true)
|-- PLAN_CVRG_MOS_NUM: short (nullable = true)
|-- SP_ALZHDMTA: short (nullable = true)
|-- SP_CHF: short (nullable = true)
|-- SP_CHRNKIDN: short (nullable = true)
|-- SP_CNCR: short (nullable = true)
|-- SP_COPD: short (nullable = true)
|-- SP_DEPRESSN: short (nullable = true)
|-- SP_DIABETES: short (nullable = true)
|-- SP_ISCHMCHT: short (nullable = true)
|-- SP_OSTEOPRS: short (nullable = true)
|-- SP_RA_OA: short (nullable = true)
|-- SP_STRKETIA: short (nullable = true)
|-- MEDREIMB_IP: double (nullable = true)
|-- BENRES_IP: double (nullable = true)
|-- PPPYMT_IP: double (nullable = true)
|-- MEDREIMB_OP: double (nullable = true)
|-- BENRES_OP: double (nullable = true)
|-- PPPYMT_OP: double (nullable = true)
|-- MEDREIMB_CAR: double (nullable = true)
|-- BENRES_CAR: double (nullable = true)
|-- PPPYMT_CAR: double (nullable = true)
```


APPENDIX

DATA SCHEMA

Inpatient Claims

Filename: Multiple Files

Format: Parquet

Size: 1,332,822 records, 75.3 MB

```
root
|-- DESYNPUF_ID: string (nullable = true)
|-- CLM_ID: long (nullable = true)
|-- SEGMENT: short (nullable = true)
|-- CLM_FROM_DT: date (nullable = true)
|-- CLM_THRU_DT: date (nullable = true)
|-- PRVDR_NUM: string (nullable = true)
|-- CLM_PMT_AMT: double (nullable = true)
|-- NCH_PMRRY_PYR_CLM_PD_AMT: double (nullable = true)
|-- AT_PHYSN_NPI: string (nullable = true)
|-- OP_PHYSN_NPI: string (nullable = true)
|-- OT_PHYSN_NPI: string (nullable = true)
|-- CLM_ADMSN_DT: date (nullable = true)
|-- ADMING_ICD9_DGNS_CD: string (nullable = true)
|-- CLM_PASS_THRU_PER_DIEM_AMT: double (nullable = true)
|-- NCH_BENE_IP_DDCTBL_AMT: double (nullable = true)
|-- NCH_BENE_PTA_COINSRNC_LBLTY_AM: double (nullable = true)
|-- NCH_BENE_BLOOD_DDCTBL_LBLTY_AM: double (nullable = true)
|-- CLM_UTILZTN_DAY_CNT: short (nullable = true)
|-- NCH_BENE_DSCHRG_DT: date (nullable = true)
|-- CLM_DRG_CD: string (nullable = true)
|-- ICD9_DGNS_CD_1: string (nullable = true)
|-- ICD9_DGNS_CD_2: string (nullable = true)
|-- ICD9_DGNS_CD_3: string (nullable = true)
|-- ICD9_DGNS_CD_4: string (nullable = true)
|-- ICD9_DGNS_CD_5: string (nullable = true)
|-- ICD9_DGNS_CD_6: string (nullable = true)
|-- ICD9_DGNS_CD_7: string (nullable = true)
|-- ICD9_DGNS_CD_8: string (nullable = true)
|-- ICD9_DGNS_CD_9: string (nullable = true)
|-- ICD9_DGNS_CD_10: string (nullable = true)
|-- ICD9_PRCDR_CD_1: string (nullable = true)
|-- ICD9_PRCDR_CD_2: string (nullable = true)
|-- ICD9_PRCDR_CD_3: string (nullable = true)
|-- ICD9_PRCDR_CD_4: string (nullable = true)
|-- ICD9_PRCDR_CD_5: string (nullable = true)
|-- ICD9_PRCDR_CD_6: string (nullable = true)
|-- HCPCS_CD_1: string (nullable = true)
|-- HCPCS_CD_2: string (nullable = true)
|-- HCPCS_CD_3: string (nullable = true)
|-- HCPCS_CD_4: string (nullable = true)
|-- HCPCS_CD_5: string (nullable = true)
|-- HCPCS_CD_6: string (nullable = true)
|-- HCPCS_CD_7: string (nullable = true)
|-- HCPCS_CD_8: string (nullable = true)
|-- HCPCS_CD_9: string (nullable = true)
|-- HCPCS_CD_10: string (nullable = true)
|-- HCPCS_CD_11: string (nullable = true)
|-- HCPCS_CD_12: string (nullable = true)
|-- HCPCS_CD_13: string (nullable = true)
|-- HCPCS_CD_14: string (nullable = true)
|-- HCPCS_CD_15: string (nullable = true)
|-- HCPCS_CD_16: string (nullable = true)
|-- HCPCS_CD_17: string (nullable = true)
|-- HCPCS_CD_18: string (nullable = true)
|-- HCPCS_CD_19: string (nullable = true)
|-- HCPCS_CD_20: string (nullable = true)
|-- HCPCS_CD_21: string (nullable = true)
|-- HCPCS_CD_22: string (nullable = true)
|-- HCPCS_CD_23: string (nullable = true)
|-- HCPCS_CD_24: string (nullable = true)
|-- HCPCS_CD_25: string (nullable = true)
|-- HCPCS_CD_26: string (nullable = true)
|-- HCPCS_CD_27: string (nullable = true)
|-- HCPCS_CD_28: string (nullable = true)
|-- HCPCS_CD_29: string (nullable = true)
|-- HCPCS_CD_30: string (nullable = true)
|-- HCPCS_CD_31: string (nullable = true)
|-- HCPCS_CD_32: string (nullable = true)
|-- HCPCS_CD_33: string (nullable = true)
|-- HCPCS_CD_34: string (nullable = true)
|-- HCPCS_CD_35: string (nullable = true)
|-- HCPCS_CD_36: string (nullable = true)
|-- HCPCS_CD_37: string (nullable = true)
|-- HCPCS_CD_38: string (nullable = true)
|-- HCPCS_CD_39: string (nullable = true)
|-- HCPCS_CD_40: string (nullable = true)
|-- HCPCS_CD_41: string (nullable = true)
|-- HCPCS_CD_42: string (nullable = true)
|-- HCPCS_CD_43: string (nullable = true)
|-- HCPCS_CD_44: string (nullable = true)
|-- HCPCS_CD_45: string (nullable = true)
```

APPENDIX

DATA SCHEMA

Outpatient Claims

Filename: Multiple Files

Format: Parquet

Size: 15,826,987 records, 667.6 MB

```
root
|-- DESYNPUF_ID: string (nullable = true)
|-- CLM_ID: long (nullable = true)
|-- SEGMENT: short (nullable = true)
|-- CLM_FROM_DT: date (nullable = true)
|-- CLM_THRU_DT: date (nullable = true)
|-- PRVDR_NUM: string (nullable = true)
|-- CLM_PMT_AMT: double (nullable = true)
|-- NCH_PMRY_PYR_CLM_PD_AMT: double (nullable = true)
|-- AT_PHYSN_NPI: string (nullable = true)
|-- OP_PHYSN_NPI: string (nullable = true)
|-- OT_PHYSN_NPI: string (nullable = true)
|-- NCH_BENE_BLOOD_DDCTBL_LBLTY_AM: double (nullable = true)
|-- ICD9_DGNS_CD_1: string (nullable = true)
|-- ICD9_DGNS_CD_2: string (nullable = true)
|-- ICD9_DGNS_CD_3: string (nullable = true)
|-- ICD9_DGNS_CD_4: string (nullable = true)
|-- ICD9_DGNS_CD_5: string (nullable = true)
|-- ICD9_DGNS_CD_6: string (nullable = true)
|-- ICD9_DGNS_CD_7: string (nullable = true)
|-- ICD9_DGNS_CD_8: string (nullable = true)
|-- ICD9_DGNS_CD_9: string (nullable = true)
|-- ICD9_DGNS_CD_10: string (nullable = true)
|-- ICD9_PRCDR_CD_1: string (nullable = true)
|-- ICD9_PRCDR_CD_2: string (nullable = true)
|-- ICD9_PRCDR_CD_3: string (nullable = true)
|-- ICD9_PRCDR_CD_4: string (nullable = true)
|-- ICD9_PRCDR_CD_5: string (nullable = true)
|-- ICD9_PRCDR_CD_6: string (nullable = true)
|-- NCH_BENE_PTB_DDCTBL_AMT: double (nullable = true)
|-- NCH_BENE_PTB_COINSRNC_AMT: double (nullable = true)
|-- ADMTNG_ICD9_DGNS_CD: string (nullable = true)
|-- HCPCS_CD_1: string (nullable = true)
|-- HCPCS_CD_2: string (nullable = true)
|-- HCPCS_CD_3: string (nullable = true)
|-- HCPCS_CD_4: string (nullable = true)
|-- HCPCS_CD_5: string (nullable = true)
|-- HCPCS_CD_6: string (nullable = true)
|-- HCPCS_CD_7: string (nullable = true)
|-- HCPCS_CD_8: string (nullable = true)
|-- HCPCS_CD_9: string (nullable = true)
|-- HCPCS_CD_10: string (nullable = true)
|-- HCPCS_CD_11: string (nullable = true)
|-- HCPCS_CD_12: string (nullable = true)
|-- HCPCS_CD_13: string (nullable = true)
|-- HCPCS_CD_14: string (nullable = true)
|-- HCPCS_CD_15: string (nullable = true)
|-- HCPCS_CD_16: string (nullable = true)
|-- HCPCS_CD_17: string (nullable = true)
|-- HCPCS_CD_18: string (nullable = true)
|-- HCPCS_CD_19: string (nullable = true)
|-- HCPCS_CD_20: string (nullable = true)
|-- HCPCS_CD_21: string (nullable = true)
|-- HCPCS_CD_22: string (nullable = true)
|-- HCPCS_CD_23: string (nullable = true)
|-- HCPCS_CD_24: string (nullable = true)
|-- HCPCS_CD_25: string (nullable = true)
|-- HCPCS_CD_26: string (nullable = true)
|-- HCPCS_CD_27: string (nullable = true)
|-- HCPCS_CD_28: string (nullable = true)
|-- HCPCS_CD_29: string (nullable = true)
|-- HCPCS_CD_30: string (nullable = true)
|-- HCPCS_CD_31: string (nullable = true)
|-- HCPCS_CD_32: string (nullable = true)
|-- HCPCS_CD_33: string (nullable = true)
|-- HCPCS_CD_34: string (nullable = true)
|-- HCPCS_CD_35: string (nullable = true)
|-- HCPCS_CD_36: string (nullable = true)
|-- HCPCS_CD_37: string (nullable = true)
|-- HCPCS_CD_38: string (nullable = true)
|-- HCPCS_CD_39: string (nullable = true)
|-- HCPCS_CD_40: string (nullable = true)
|-- HCPCS_CD_41: string (nullable = true)
|-- HCPCS_CD_42: string (nullable = true)
|-- HCPCS_CD_43: string (nullable = true)
|-- HCPCS_CD_44: string (nullable = true)
|-- HCPCS_CD_45: string (nullable = true)
```

APPENDIX

DATA SCHEMA

Prescription Drug Events

Filename: Multiple Files

Format: Parquet

Size: 111,085,969 records, 2.2 GB

```
root
|-- DESYNPUF_ID: string (nullable = true)
|-- PDE_ID: long (nullable = true)
|-- SRVC_DT: date (nullable = true)
|-- PROD_SRVC_ID: string (nullable = true)
|-- QTY_DSPNSD_NUM: short (nullable = true)
|-- DAYS_SUPLY_NUM: short (nullable = true)
|-- PTNT_PAY_AMT: double (nullable = true)
|-- TOT_RX_CST_AMT: double (nullable = true)
```

APPENDIX S3 STORAGE

The screenshot displays the AWS S3 console interface. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', and a user profile icon. The right side of the header shows a notification bell, the account name 'de-projects', the region 'Global', and a 'Support' link.

The main content area is titled 'Amazon S3' and 'cms-data-1'. Below this, there are tabs for 'Overview', 'Properties', 'Permissions', and 'Management'. A search bar is present with the placeholder text 'Type a prefix and press Enter to search. Press ESC to clear.' Below the search bar are buttons for 'Upload', 'Create folder', 'Download', and 'Actions'.

A modal window titled 'Get size' is open in the center. It displays the following information:

- Selection:** 0 Objects, 5 Folders
- Total size:** 3.3 GB
- Total objects:** 1425

The modal lists the following folders and their contents:

- Annual_Cost_Predictions/**
3 Objects - 188.0 MB
- Beneficiary_Summary/**
119 Objects - 188.9 MB
- Inpatient_Claims/**
21 Objects - 75.3 MB
- Outpatient_Claims/**
161 Objects - 667.6 MB
- Prescription_Drug_Events/**
1121 Objects - 2.2 GB

At the bottom of the modal are 'Cancel' and 'Done' buttons.

In the background, the 'cms-data-1' bucket is visible. It shows a list of objects with columns for 'Name', 'Size', 'Storage class', and 'Actions'. The objects listed are:

- Annual_Cost_Predictions
- Beneficiary_Summary
- Inpatient_Claims
- Outpatient_Claims
- Prescription_Drug_Events

The region is set to 'US East (N. Virginia)' and the storage class is 'Standard'. The view is set to 'Grid' and shows 'Viewing 1 to 5' objects.

APPENDIX REDSHIFT

Resource Groups

Cluster `redshift-cms-cluster`

Database `dev`

Database user `awsuser`

Schema

cms

Tables

Showing 5 of 5 table(s)

Filter tables

▶

`annual_cost_predictions`

▶

`beneficiary_summary`

▶

`inpatient_claims`

▶

`outpatient_claims`

▶

`prescription_drug_eve...`

create-copy-c...

```
--
70 --ML Predictions on Annual Cost--
71 create table cms.annual_cost_predictions (ANNUAL_COST float, PATIENT_ID varchar, AGE_YRS float, GENDER int2, RACE int2, STATE int2, DX int,
72     PX int, HCPCS int, SP_ALZHDMTA int2, SP_CHF int2, SP_CHRNKIDN int2, SP_CNCR int2,
73     SP_COPD int2, SP_DEPRESSN int2, SP_DIABETES int2, SP_ISCHMCHT int2, SP_OSTEOPRS int2,
74     SP_RA_OA int2, SP_STRKETIA int2, GENDER_index float, RACE_index float, STATE_index float,
75     SP_ALZHDMTA_index float, SP_CHF_index float, SP_CHRNKIDN_index float,
76     SP_CNCR_index float, SP_COPD_index float, SP_DEPRESSN_index float,
77     SP_DIABETES_index float, SP_ISCHMCHT_index float, SP_OSTEOPRS_index float,
78     SP_RA_OA_index float, SP_STRKETIA_index float, features varchar, prediction float)
79
80 --Copy Commands into Redshift--
81 copy cms.beneficiary_summary
82 from 's3://cms-data-1/Beneficiary_Summary/'
83 IAM_ROLE 'arn:aws:iam::799687528413:role/RedshiftDWS3FullAccess'
```

Run query

Save as

Save

Clear

Use Ctrl + Enter to run query, Ctrl + Space to autocomplete

[Send feedback](#)

APPENDIX

LAMBDA REDSHIFT LOAD

```
--Beneficiary Summary--  
set search_path to cms;  
begin;  
delete from cms.beneficiary_summary;  
copy cms.beneficiary_summary  
from 's3://cms-data-1/Beneficiary_Summary/'  
IAM_ROLE  
'arn:aws:iam::799687528413:role/RedshiftDWS3FullAccess'  
FORMAT AS PARQUET;  
commit;
```