M12 Practical Challenge: Building a Supervised Learning Model via Amazon SageMaker Studio GUI

In this project, I choose data from Kaggle, and the data is about Heart Failure Prediction Dataset. (https://www.kaggle.com/fedesoriano/heart-failure-prediction). We have several features such as Age, Sex, ChestPainType, RestingBP, Cholesterol, FastingBS, RestingECG, MaxHR, ExerciseAngina, Oldpeak, ST_Slope and target HeartDisease in this dataset.

I uploaded it into GitHub and can download it by pandas. (https://raw.githubusercontent.com/NewThread-ZY/AIM-5014-100/main/heart.csv).

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37	M	ATA	130	283	0	ST	98	N	0.0	Up	0
3	48	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1
4	54	M	NAP	150	195	0	Normal	122	N	0.0	Up	0
...	...	...	...	...	...	...	...	...	...	...	...	...
913	45	M	TA	110	264	0	Normal	132	N	1.2	Flat	1
914	68	M	ASY	144	193	1	Normal	141	N	3.4	Flat	1
915	57	M	ASY	130	131	0	Normal	115	Y	1.2	Flat	1
916	57	F	ATA	130	236	0	LVH	174	N	0.0	Flat	1
917	38	M	NAP	138	175	0	Normal	173	N	0.0	Up	0

918 rows × 12 columns

Our data consists of some numerical data and category data. We change it from category data to numerical data and normalization them.

```
data.Sex = data.Sex.replace({'M': 0, 'F': 1})
data.ChestPainType = data.ChestPainType.replace({'ATA': 0, 'NAP': 1,'ASY': 2,'TA': 3})
data.RestingECG = data.RestingECG.replace({'Normal': 0, 'ST': 1,'LVH': 2})
data.ExerciseAngina = data.ExerciseAngina.replace({'N': 0, 'Y': 1})
data.ST_Slope = data.ST_Slope.replace({'Up': 0, 'Flat': 1, 'Down': 2})
data
```

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	0	0	140	289	0	0	172	0	0.0	0	0
1	49	1	1	160	180	0	0	156	0	1.0	1	1
2	37	0	0	130	283	0	1	98	0	0.0	0	0
3	48	1	2	138	214	0	0	108	1	1.5	1	1
4	54	0	1	150	195	0	0	122	0	0.0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...
913	45	0	3	110	264	0	0	132	0	1.2	1	1
914	68	0	2	144	193	1	0	141	0	3.4	1	1
915	57	0	2	130	131	0	0	115	1	1.2	1	1
916	57	1	0	130	236	0	2	174	0	0.0	1	1
917	38	0	1	138	175	0	0	173	0	0.0	0	0

918 rows × 12 columns

```
data = (data-data.min())/(data.max()-data.min())
data
```


```
data = (data-data.min())/(data.max()-data.min())
data
```


	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	0.244898	0.0	0.000000	0.70	0.479270	0.0	0.0	0.788732	0.0	0.295455	0.0	0.0
1	0.428571	1.0	0.333333	0.80	0.298507	0.0	0.0	0.676056	0.0	0.409091	0.5	1.0
2	0.183673	0.0	0.000000	0.65	0.469320	0.0	0.5	0.267606	0.0	0.295455	0.0	0.0
3	0.408163	1.0	0.666667	0.69	0.354892	0.0	0.0	0.338028	1.0	0.465909	0.5	1.0
4	0.530612	0.0	0.333333	0.75	0.323383	0.0	0.0	0.436620	0.0	0.295455	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...
913	0.346939	0.0	1.000000	0.55	0.437811	0.0	0.0	0.507042	0.0	0.431818	0.5	1.0
914	0.816327	0.0	0.666667	0.72	0.320066	1.0	0.0	0.570423	0.0	0.681818	0.5	1.0
915	0.591837	0.0	0.666667	0.65	0.217247	0.0	0.0	0.387324	1.0	0.431818	0.5	1.0
916	0.591837	1.0	0.000000	0.65	0.391376	0.0	1.0	0.802817	0.0	0.295455	0.5	1.0
917	0.204082	0.0	0.333333	0.69	0.290216	0.0	0.0	0.795775	0.0	0.295455	0.0	0.0

918 rows × 12 columns

Although we didn’t require to do this and it also can run in the model, we can have a 0.01% improvement in accuracy if we did it by ourselves. Our targets 1 and 0 will be predicted by age, sex, and some other features.

We can know that there are 918 rows and 12 columns on this dataset.

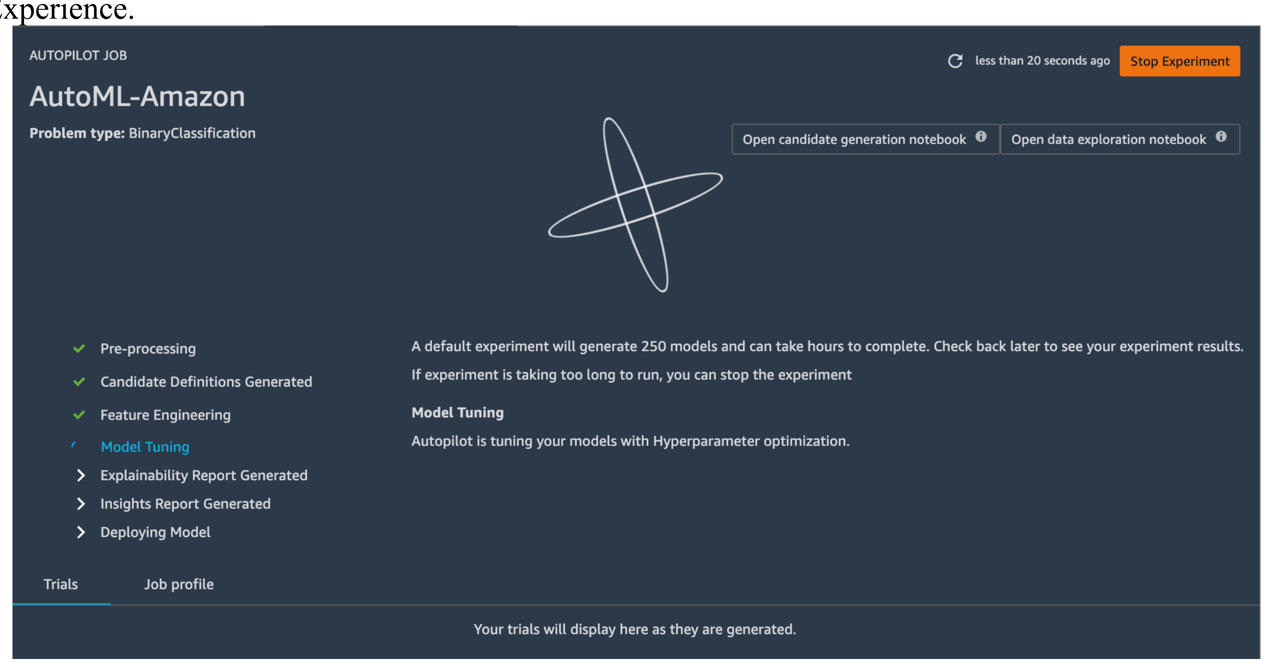
```
train_data, test_data, _ = np.split(data.sample(frac = 1, random_state = 42), [int(0.8*len(data)), int(len(data))])
```

```
#save to csv
train_data.to_csv('./heart/AutoML_train.csv', index = False, header = True)
test_data.to_csv('./heart/AutoML_test.csv', index = False, header = True)
```

I spat the dataset into the training and testing dataset. The training dataset occupies 80% of this dataset.

```
import sagemaker

prefix = 'sagemaker/DEMO-autopilot/input'
sess   = sagemaker.Session()

uri = sess.upload_data(path="./heart/AutoML_train.csv", key_prefix=prefix)

print(uri)

s3://sagemaker-us-east-1-669717135279/sagemaker/DEMO-autopilot/input/AutoML_train.csv
```

Then I upload the training dataset to Amazon S3 and use this Uri to create Autopilot Experience.



AUTOPILOT JOB

less than a minute ago Deploy model

AutoML-Amazon

Problem type: BinaryClassification

Open candidate generation notebook | Open data exploration notebook

Best model endpoint: An error occurred deploying the best model.

Trials | Job profile

0 rows selected

Best Model	Model name	Status	Start time	Objective: F1_binary	F1	Accuracy
Best Model	AutoML-AmazonDuzU...	Completed	1 hour ago	0.863	*0.863*	*0.883*
	AutoML-AmazonDuzU...	Completed	1 hour ago	0.859	*0.859*	*0.881*
	AutoML-AmazonDuzU...	Completed	39 minutes ago	0.858	*0.858*	*0.88*
	AutoML-AmazonDuzU...	Completed	55 minutes ago	0.858	*0.858*	*0.88*
	AutoML-AmazonDuzU...	Completed	1 hour ago	0.858	*0.858*	*0.88*
	AutoML-AmazonDuzU...	Completed	16 minutes ago	0.857	*0.857*	*0.88*
	AutoML-AmazonDuzU...	Completed	40 minutes ago	0.857	*0.857*	*0.88*
	AutoML-AmazonDuzU...	Completed	22 minutes ago	0.856	*0.856*	*0.88*
	AutoML-AmazonDuzU...	Completed	58 minutes ago	0.855	*0.855*	*0.88*
	AutoML-AmazonDuzU...	Completed	1 hour ago	0.858	*0.858*	*0.879*
	AutoML-AmazonDuzU...	Completed	24 minutes ago	0.858	*0.858*	*0.879*
	AutoML-AmazonDuzU...	Completed	1 hour ago	0.856	*0.856*	*0.879*
	AutoML-AmazonDuzU...	Completed	1 hour ago	0.856	*0.856*	*0.879*
	AutoML-AmazonDuzU...	Completed	50 minutes ago	0.855	*0.855*	*0.879*

automl-ama

Then I click open the data exploration notebook.

Amazon SageMaker Autopilot Data Exploration Report ¶

This report contains insights about the dataset you provided as input to the AutoML job. This data report was generated by **AutoML-Amazon** AutoLM job. To check for any issues with your data and possible improvements that can be made to it, consult the sections below for guidance. You can use information about the predictive power of each feature in the **Data Sample** section and from the correlation matrix in the **Cross Column Statistics** section to help select a subset of the data that is most significant for making predictions.

Note: SageMaker Autopilot data reports are subject to change and updates. It is not recommended to parse the report using automated tools, as they may be impacted by such changes.

Dataset Summary

Dataset Properties

Rows	Columns	Duplicate rows	Target column	Missing target values	Invalid target values	Detected problem type
734	12	0.00%	HeartDisease	0.00%	0.00%	BinaryClassification

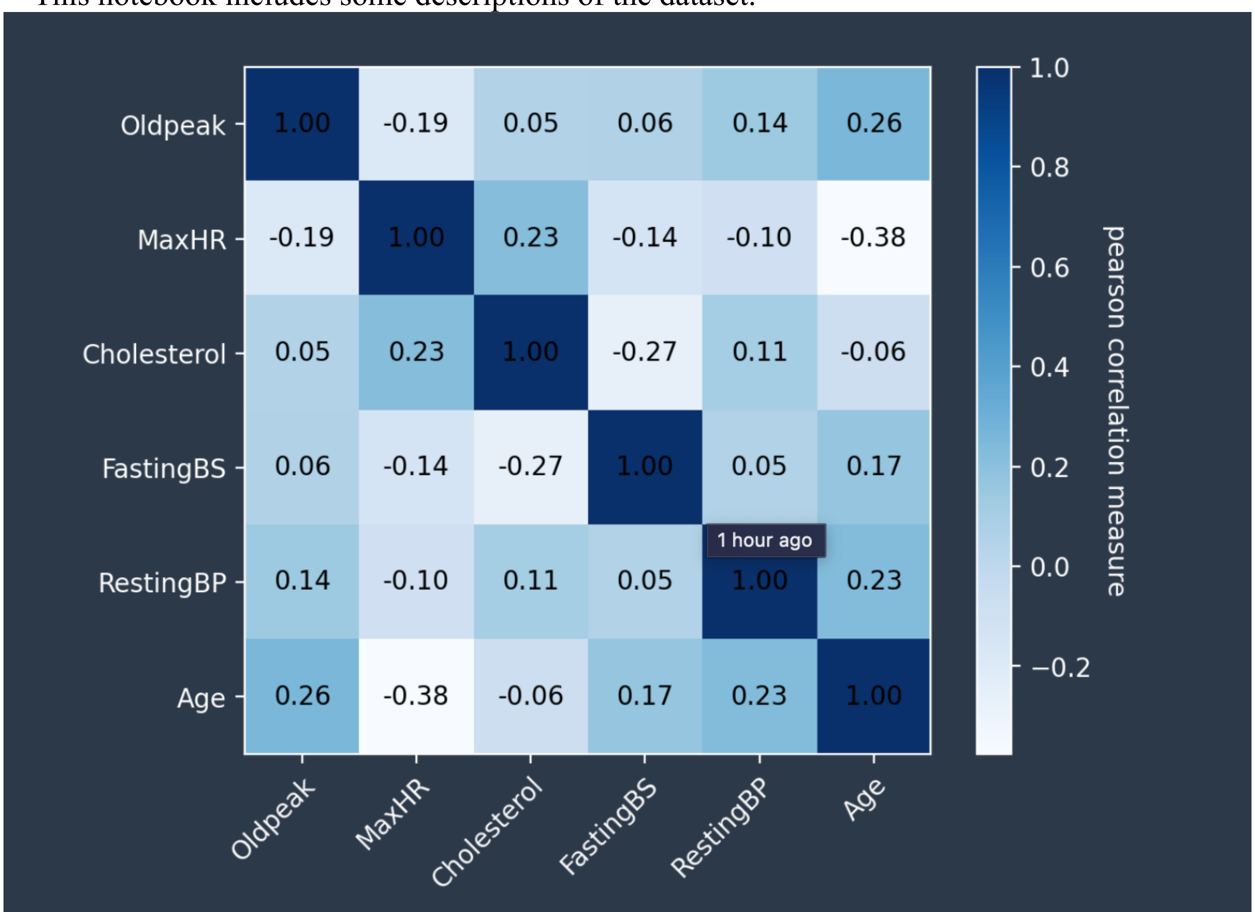
Detected Column Types

	Numeric	Categorical	Text	Datetime	Sequence
Column Count	6	5	0	0	0
Percentage	54.55%	45.45%	0.00%	0.00%	0.00%

Report Contents

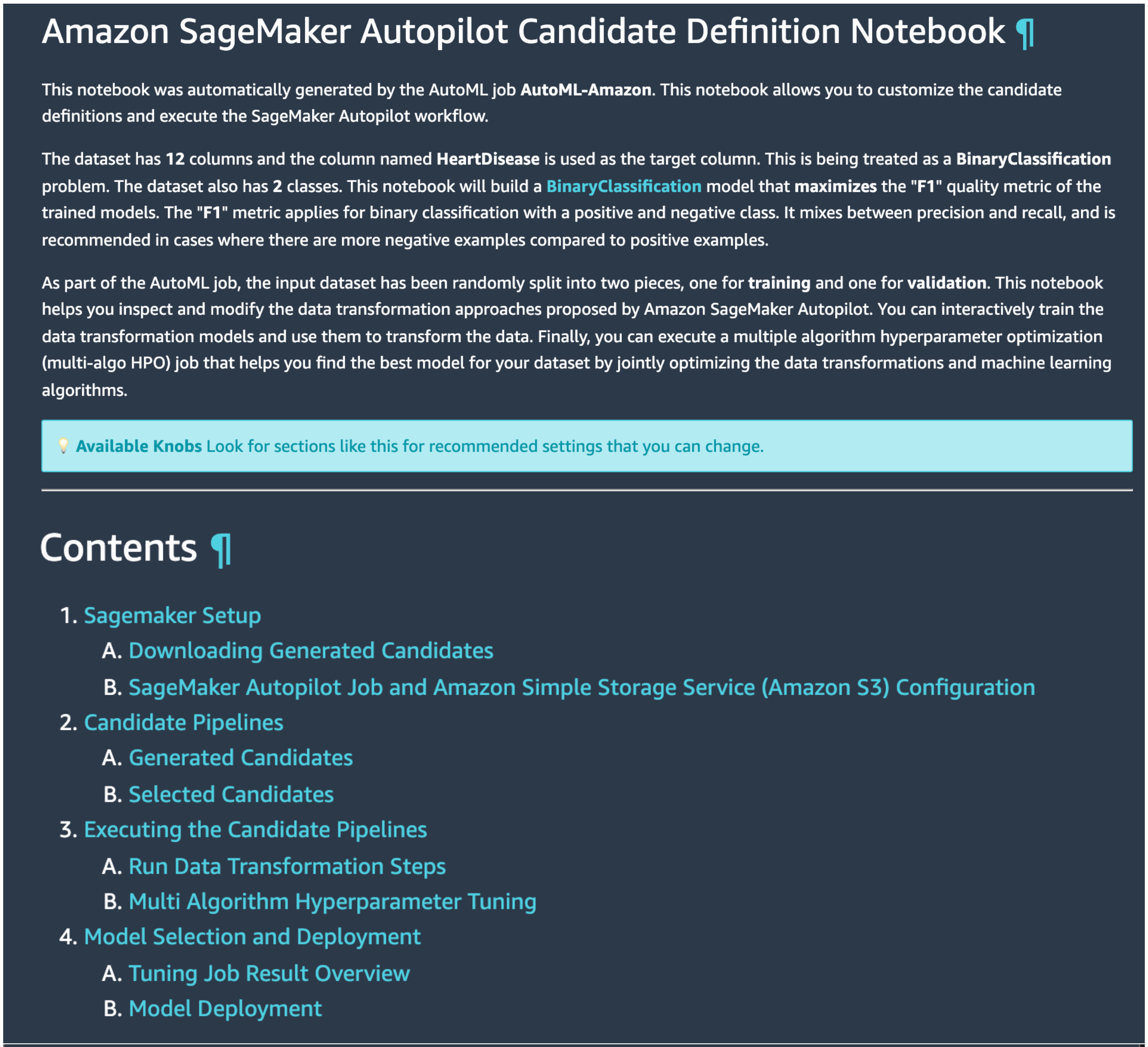
1. Target Analysis
2. Data Sample
3. Duplicate Rows
4. Cross Column Statistics
5. Anomalous Rows
6. Missing Values
7. Cardinality
8. Descriptive Stats
9. Definitions

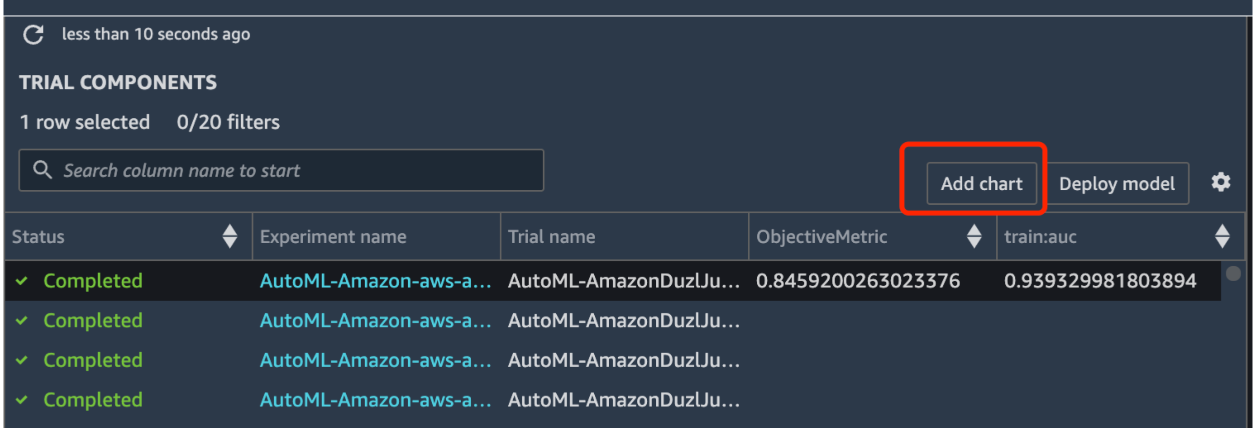
This notebook includes some descriptions of the dataset.



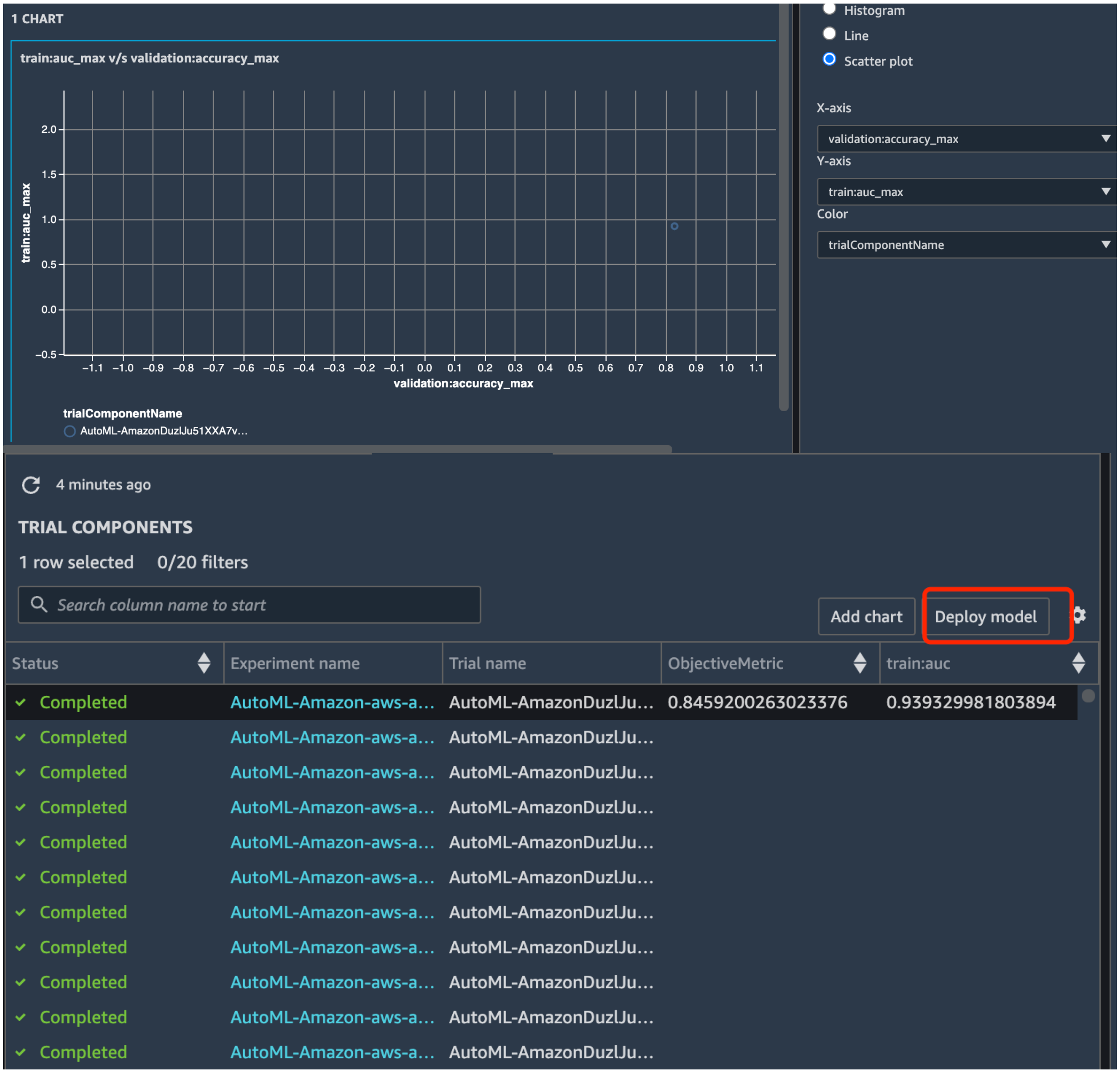
Cross column correlation for numeric features

And I click open candidate generation notebook.

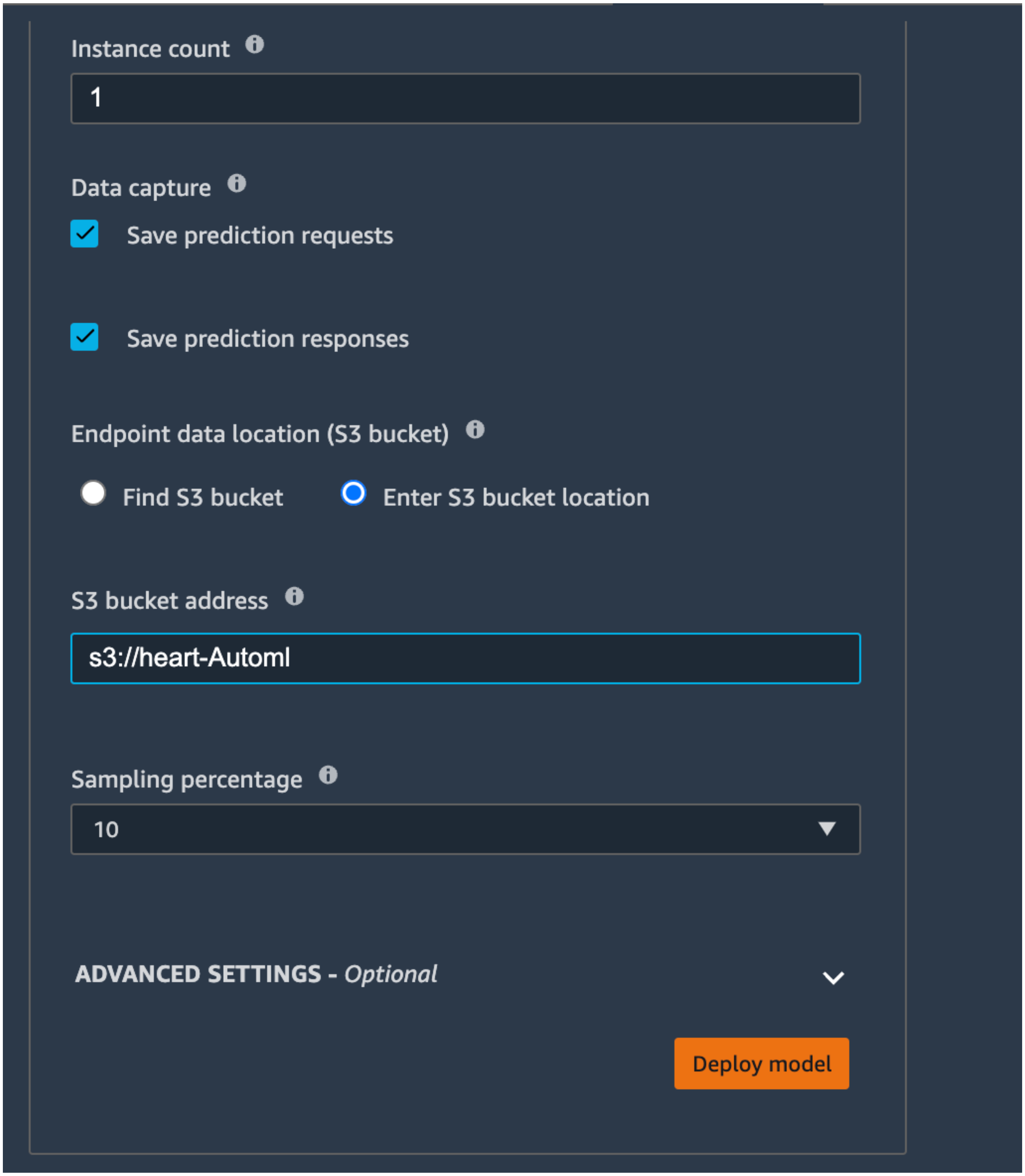




Then I chose the highest accuracy one and click add chart.



And then I click deploy the model.



Here is what I type.

Then I used this model to do predict and calculate some accuracy indicators.

```
import boto3,sys
sm_rt = boto3.Session().client('runtime.sagemaker')
```

```
tp=tn=fp=fn=count=0
with open('./heart/AutoML_test.csv') as f:
    lines = f.readlines()
    for l in lines[1:]:
        l = l.split(',')
        label = l[-1]
        l = l[:-1]
        l = ','.join(l)
        response = sm_rt.invoke_endpoint(EndpointName=ep_name,ContentType='text/csv',Accept='text/csv',Body=l)
        response = response['Body'].read().decode('utf-8')
        if '1' in label:
            if '1' in response:
                tp+=1
            else:
                fn+=1
        if '0' in label:
            if '0' in response:
                tn+=1
            else:
                fp+=1
        count+=1
        if (count%100==0):
            sys.stdout.write(str(count)+' ')
print('done')
```

```
100 done
```

```
print("%d %d" %(tp, fn))
print("%d %d" %(fp, tn))
```

```
88 4
0 184
```

```
accuracy = (tp+tn)/(tp+tn+fn+fp)
accuracy
```

```
0.9855072463768116
```

```
precision = tp/(tp+fp)
precision
```

```
1.0
```

```
recall = tn/(tn+fn)
recall
```

```
0.9787234042553191
```

```
f1 = (2*precision*recall)/(precision+recall)
f1
```

```
0.989247311827957
```

Here we successfully used Amazon Sagemaker to train a model and use it to do predict.

In SageMaker, I think it's more flexible than DataRobot. We can use a notebook to do some action by ourselves. But in DataRobots, you just only can combine it with Alteryx. And in SageMaker, I think it's easier to deploy the model. In a word, I prefer to SageMaker.