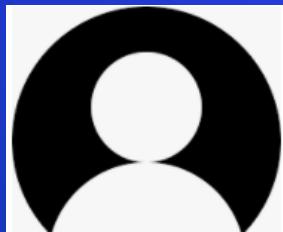


# Data Science Final Project Presentation

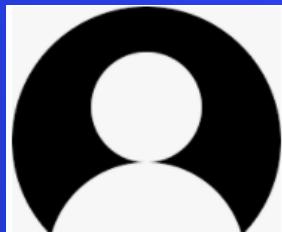
Class: IBM IMVAI-1901  
Team 2



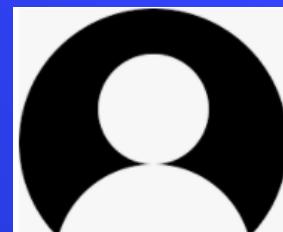
# Team Members



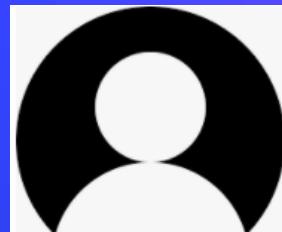
Allen Vincent  
Hui Kim Seng



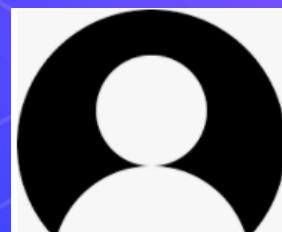
Aung Pye Phyo



Chong Yul Chin  
(Eunice)



Law Tuan Lee  
(Jolene)



Lai Wei Jie

# Agenda

Team Members	Roles & Responsibilities	Agenda
Eunice Chong	<p>Product Owner</p> <p>Plans, defines the problem, builds hypothesis, and provides feedback</p>	<p>1) Introduction</p> <p>2) Planning Analytics</p>
Allen Vincent Hui Kim Seng	<p>Data Analyst / Data Engineer</p> <p>Collects and analyzes data from operational systems and databases (e.g Kaggle, data.gov.sg) ; understands data and uses statistical method and analytical tool to interpret the data</p> <p>Uses computational notebook to perform data cleansing, and creates data visualization to make information more accessible and understandable</p>	<p>3) Descriptive Analytics</p> <p>4) Diagnostic Analytics</p>
Law Tuan Lee (Jolene)	<p>Data Scientist</p> <p>Manages data by building, testing and transforming raw data into usable pipeline for machine learning.</p> <p>Extracts insights from data patterns; builds and tests the model to provide a prediction</p>	<p>5) Predictive Analytics</p>

# Agenda

Team Members	Roles & Responsibilities	Agenda
Lai Wei Jie	<p>Data Scientist</p> <p>Takes inputs from prediction to deploy predictive models via machine learning. Make sure a satisfactory model has been developed before deploying into production environment.</p> <p>Provides recommendations to our user</p>	6. Prescriptive Analytics
Aung Pye Phyo	<p>Developer</p> <p>Enables models to be deployed and develop software application effectively that leverage the insights and data gathered from other data science team.</p> <p>Actively monitor the quality of model in production to detect performance degradation and model staleness. Research and implement best practices to enhance existing machine learning infrastructure.</p>	7. Environment Check 8. Summary and Reflection

# 1. Introduction



# Project Objectives

Using Data analytics Lifecycle and following the Data Science Methodology, our objective is to accurately predict health insurance charges and provide recommendations to our Users on what they do to avoid paying a higher health insurance charges than other Users with similar biometrics to achieve greater financial savings.



# Data Analytics Lifecycle

## PLANNING ANALYTICS

### **What's your plan?**

Understand the status of your business

## PRESCRIPTIVE ANALYTICS

### **What should you do?**

Finally, use prescriptive analytics to decide what you should do to overcome operational obstacles found in your analysis.

## DESCRIPTIVE ANALYTICS

### **What happened?**

Identify what happened and determine how you're performing against the plan.

## DIAGNOSTIC ANALYTICS

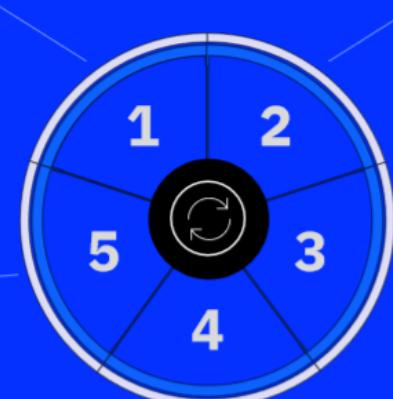
### **Why did it happen?**

Then, diagnose why problems happened and identify key patterns

## PREDICTIVE ANALYTICS

### **What happens next?**

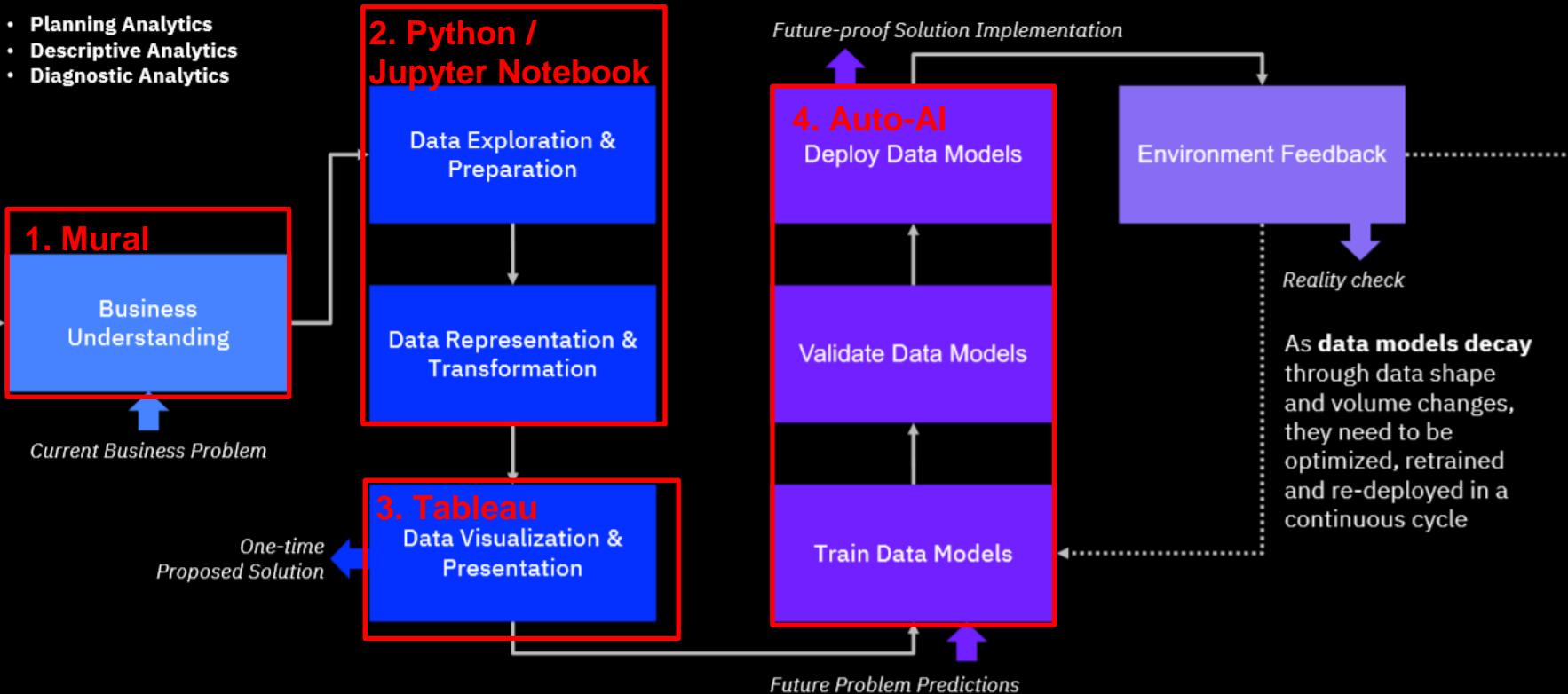
Use those patterns to predict future trends



# Tools

## Data science methodology

- Planning Analytics
- Descriptive Analytics
- Diagnostic Analytics



## 2. Planning Analytics



# Business Understanding

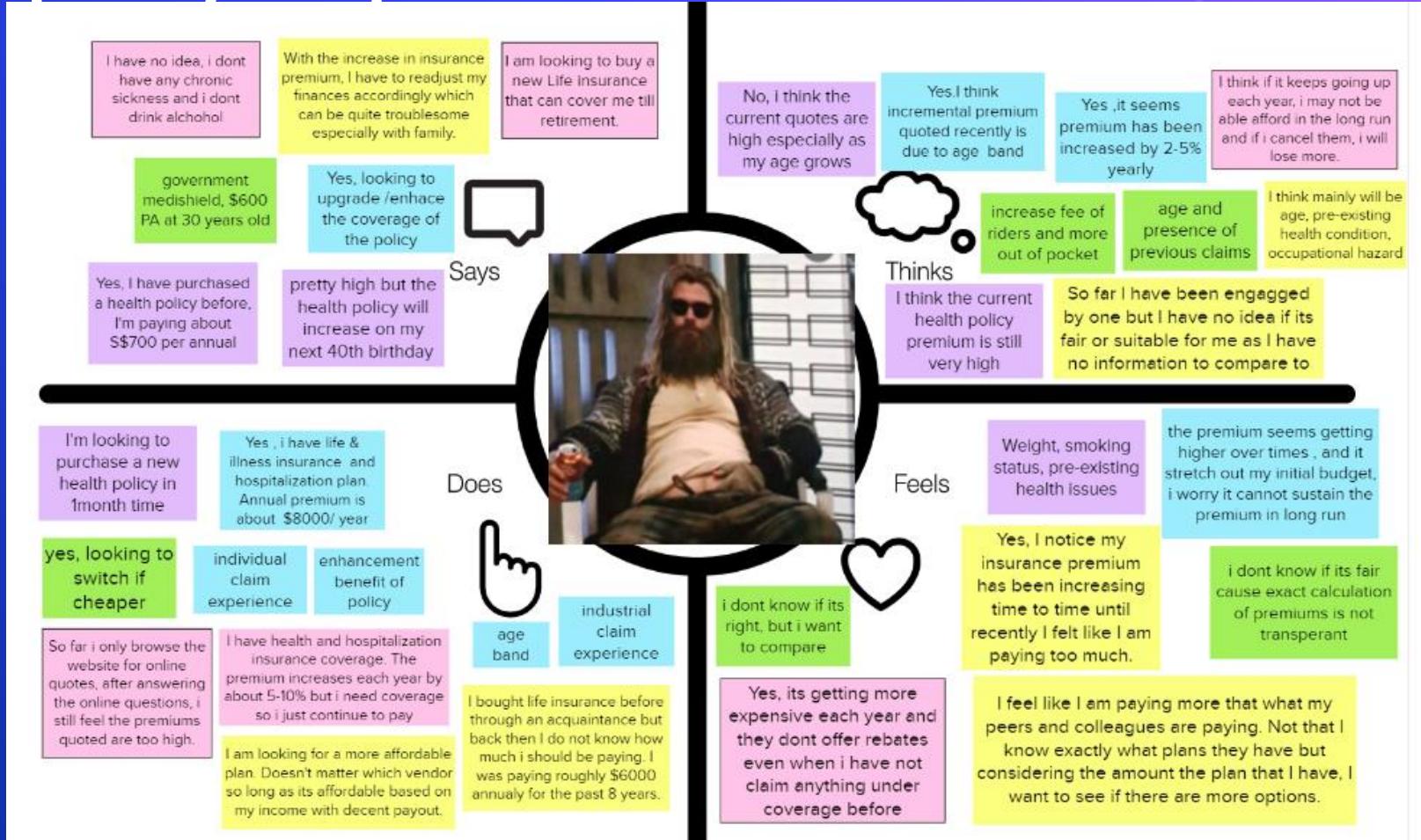
## What is the business problem?

- Why the difference in health insurance charges for different customers and what one can do to avoid paying high charges

## What is the business opportunity?

- What can health insurance company do to show transparency and promote self-discipline from customer to lead a more prudent life and be covered for the unexpected in life.

# Empathy Map



# PainPoints

- I don't know if its fair cause exact calculation of charges is not transparent
- The charges seem getting higher over times , and it has stretched my initial budget, I worry it cannot sustain the payment in long run
- Yes, it is getting more expensive each year and they don't offer rebates even when I have not claim anything under coverage before
- So far I have only browse the website for online quotes, after answering the online questions, I still feel the charges quoted are too high.

# HMW Statement

**How might we** help people to avoid paying higher annual health  
**for** insurance charges than their peers  
people looking to purchase new plans  
**so that** they can avoid being overcharged



## Reframed HMW Statement

**How might we** understand what is the expected annual health  
insurance charges  
**for** Individuals with comparable biometrics and lifestyles  
**so that** they can ease the financial burden in the long run

# Hill Statement



**Who**

Thor, a middle age working Singapore

**What**

wants to know the average health insurance charges paid by other policy holders with similar biometric and lifestyles

**Wow**

to avoid not paying more than 5% than average health insurance charges paid by policy holders with similar demographics

# Hypothesis



If we provide Thor with a tool that enables him to enter his details to generate an average health insurance charges paid by others

We will observe

- Hypothesis 1: A higher charges paid if the user is having a higher BMI (i.e. If Thor has a high BMI)
- Hypothesis 2: A lower charges paid if the user is a non-smoker (i.e. If Thor is a smoker)

# MVP

If we provide middle aged adults

With a model that enables him to compare health insurance charges paid by policy holders of similar demographics

We will address the risk of: individuals overpaying for their health insurance

By Measuring:

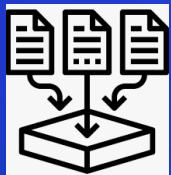
We'll know we've arrived when we observed:

- Users have a 70% successful purchase rate after getting their quote
- More than 85% download clicks rate of the recommendation generated

### 3. Descriptive Analytics



# Approach



Step 1: Gather the appropriate data set and prepare for analysis.



Step 2: Using Jupyter Notebook in Kaggle to perform

- Data Profile
- Data Cleansing
- Data Transformation



Step 3: Using Tableau to visualize preliminary cleansed data results and run summary statistics



Step 4: Create models using IBM Auto AI with Machine Learning service associated. With Auto AI, we can build and deploy a machine learning model with sophisticated training features and no coding.



Step 5: Determine the best model that can be used to predict health insurance charges based on the pipeline results and Deploy Import data using IBM Watson Studio service.



Step 6: Run Model to predict results

# Dataset Overview

## □ Data Source

<https://www.kaggle.com/datasets/simranjain17/insurance>

## □ Dataset Description

This is a basic Health Insurance Data where every row states details of one policy holder e.g. Gender, BMI and other characteristics

# Dataset - Attributes

Attributes	Description	Data Type
Age	This is the age of the person.	Integer
Sex	This is the sex of the person.	String
BMI	This is the Body Mass Index of the person where: $BMI = \text{weight} / \text{height}^2$ in metres squared.	Decimal
Children	This is the number of children of the person.	Integer
Smoker	This is a value representing whether the person is a smoker.	String
Region	This is the location of where the person stays.	String
Charges	This is the amount the person is paying for their health insurance charges annually.	Decimal

# Dataset - Attributes

Source: IBM Data Refinery

Data		Profile		Visualizations			
	age	sex	bmi	children	smoker	region	charges
	Integer	String	Decimal	Integer	String	String	Decimal
1	19	female	27.9	0	yes	southwest	16884.924
2	18	male	33.77	1	no	southeast	1725.5523
3	28	male	33	3	no	southeast	4449.462
4	33	male	22.705	0	no	northwest	21984.47061
5	32	male	28.88	0	no	northwest	3866.8552

# Data Exploration

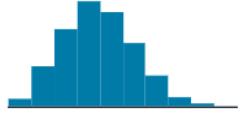
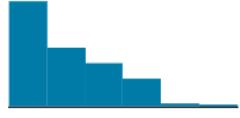
insurance.csv (55.63 kB) 001

Detail Compact Column 7 of 7 columns ▾

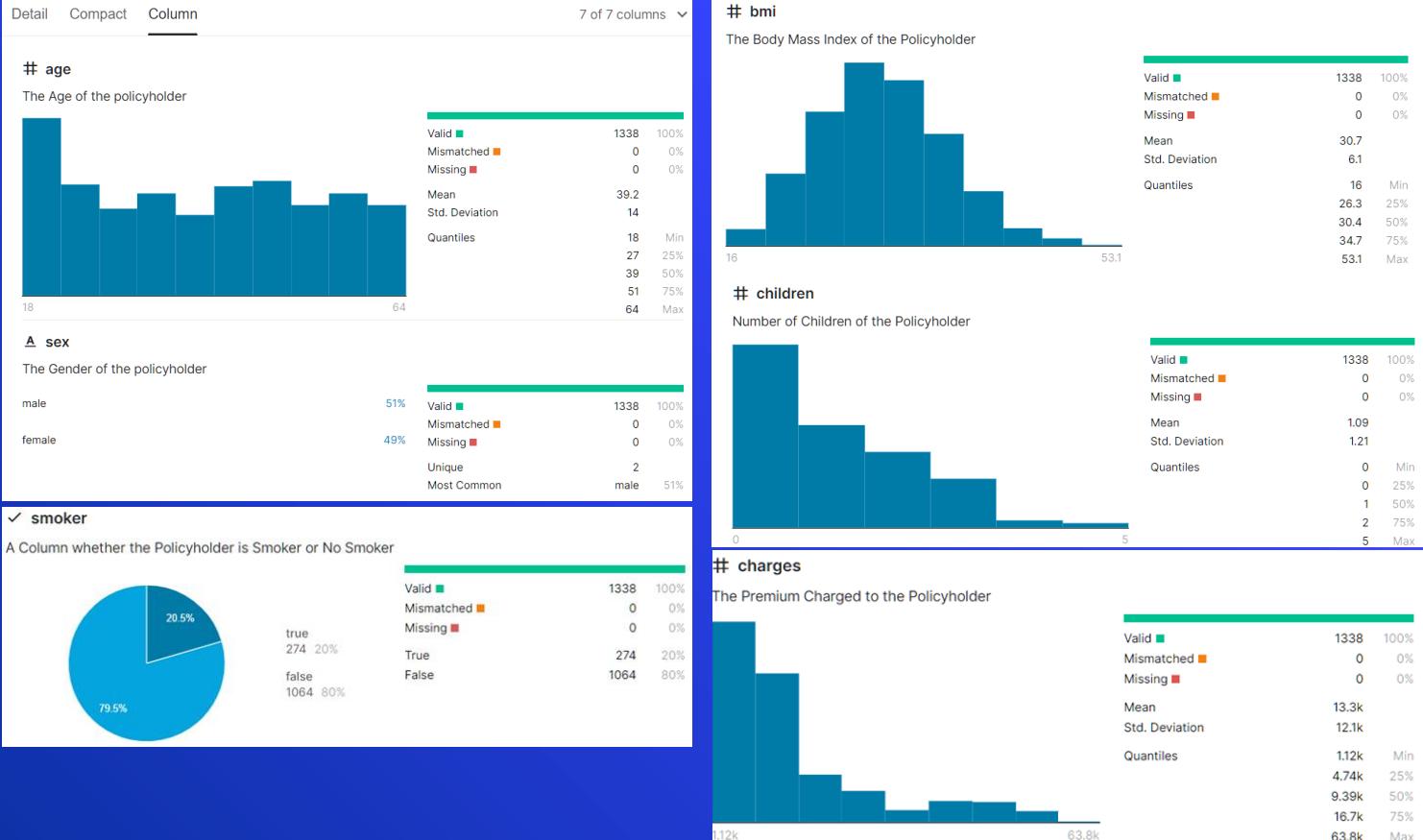
#	age	sex	bmi	children	smoker	region	charges
19		female	27.9	0	yes	southwest	16884.924
18		male	33.77	1	no	southeast	1725.5523
28		male	33	3	no	southeast	4449.462
33		male	22.705	0	no	northwest	21984.47061
32		male	28.88	0	no	northwest	3866.8552
31		female	25.74	0	no	southeast	3756.6216
46		female	33.44	1	no	southeast	8240.5896
37		female	27.74	3	no	northwest	7281.5056
37		male	29.83	2	no	northeast	6406.4107
60		female	25.84	0	no	northwest	28923.13692
25		male	26.22	0	no	northeast	2721.3208
62		female	26.29	0	yes	southeast	27808.7251

# Data Exploration

This a Basic Health Insurance Data where Every Row states details of one policyholder.

# age	# sex	# bmi	# children	✓ smoker	▲ region	# charges
The Age of the policyholder	The Gender of the policyholder	The Body Mass Index of the Policyholder	Number of Children of the Policyholder	A Column whether the Policyholder is Smoker or No Smoker	The Region where the Policyholder belongs to	The Premium Charged to the Policyholder
 18 64	male 51% female 49%	 16 53.1	 0 5	 true 274 20% false 1064 80%	southeast 27% southwest 24% Other (649) 49%	1.12k 63.8k
19	female	27.9	0	yes	southwest	16884.924
18	male	33.77	1	no	southeast	1725.5523
28	male	33	3	no	southeast	4449.462
33	male	22.705	0	no	northwest	21984.47061
32	male	28.88	0	no	northwest	3866.8552
31	female	25.74	0	no	southeast	3756.6216

# Data Exploration



# Data Preparation (Cleansing) with Python

001

## Step #0: PROBLEM STATEMENT

Insurance.csv is a data listing the health insurance charges paid by individuals.

We would like to develop a model to predict the total dollar amount that customers are willing to pay given the following attributes using Artificial Neural Networks

- Age
- Gender
- BMI
- Smoker

The model should be able to predict

- Health Insurance Premium Charges

+ Code

+ Markdown

## STEP #1: IMPORT LIBRARIES/DATASETS

Data collected from <https://www.kaggle.com/simranjain17/insurance>

+ Code

+ Markdown

# Data Preparation (Cleansing) with Python

[1]:

```
## Database Phase
import pandas as pd
import numpy as np

#Visualization Phase
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib as mpl
import matplotlib.pylab as pylab
%matplotlib inline
pd.set_option('display.max_columns', 500)
mpl.style.use('ggplot')
sns.set_style('white')
pylab.rcParams['figure.figsize'] = 12,8
```

+ Code

+ Markdown

[2]:

```
insurance = pd.read_csv('../input/insurance/insurance.csv')
insurance_copy=insurance.copy('../input/insurance/insurance.csv')
```

# Data Preparation (Cleansing) with Python

001

[3]:

```
print ("The shape of the data is (row, column):"+ str(insurance_copy.shape))
print (insurance_copy.info())
```

```
The shape of the data is (row, column):(1338, 7)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column   Non-Null Count  Dtype  
--- 
 0   age       1338 non-null   int64  
 1   sex       1338 non-null   object  
 2   bmi       1338 non-null   float64 
 3   children  1338 non-null   int64  
 4   smoker    1338 non-null   object  
 5   region    1338 non-null   object  
 6   charges   1338 non-null   float64 
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
None
```

[+ Code](#)[+ Markdown](#)

[4]:

```
df = insurance_copy
```

# Data Preparation (Cleansing) with Python

[5]:

```
insurance_csv = ('../input/insurance/insurance.csv')
insurance = pd.read_csv('../input/insurance/insurance.csv')
```

[+ Code](#)[+ Markdown](#)

[6]:

```
df.head()
```

[6]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

[7]:

```
df.tail()
```

[7]:

	age	sex	bmi	children	smoker	region	charges
1333	50	male	30.97	3	no	northwest	10600.5483
1334	18	female	31.92	0	no	northeast	2205.9808
1335	18	female	36.85	0	no	southeast	1629.8335
1336	21	female	25.80	0	no	southwest	2007.9450
1337	61	female	29.07	0	yes	northwest	29141.3603

[+ Code](#)[+ Markdown](#)

[8]:

```
insurance_copy.dtypes
```

[8]:

age		int64
sex		object
bmi		float64
children		int64
smoker		object
region		object
charges		float64
dtype:	object	

[9]:

#Checking out the statistical parameters  
insurance\_copy.describe()

[9]:

	age	bmi	children	charges
count	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	30.663397	1.094918	13270.422265
std	14.049960	6.098187	1.205493	12110.011237
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.296250	0.000000	4740.287150
50%	39.000000	30.400000	1.000000	9382.033000
75%	51.000000	34.693750	2.000000	16639.912515
max	64.000000	53.130000	5.000000	63770.428010

001

# Data Representation and Transformation with Python

Attributes	Description	Original Data Type	Transformed Data Type
Age	This is the age of the person.	Integer	Integer
Sex	This is the sex of the person.	String	Changed to Boolean in Python. 1 = Male, 0 = Female.
BMI	This is the Body Mass Index of the person where: $BMI = \text{weight} / \text{height}^2$ in metres squared.	Decimal	Decimal
Children	This is the number of children of the person.	Integer	Integer
Smoker	This is a value representing whether the person is a smoker.	String	Changed to Boolean in Python. 1 = Yes, 0 = No
Region	This is the location of where the person stays.	String	String
Charges	This is the amount the person is paying for their health insurance charges annually.	Decimal	Decimal

1-Hot Encoding

# Data Representation and Transformation with Python

```
#Checking out the categories and their respective counts in each feature
print("sex:",insurance_copy.sex.value_counts())
print("-"*40)
print("smoker:",insurance_copy.smoker.value_counts())
print("-"*40)
print("age:",insurance_copy.age.value_counts())
print("-"*40)
print("bmi:",insurance_copy.bmi.value_counts())
print("-"*40)
print("children:",insurance_copy.children.value_counts())
print("-"*40)
print("charges:",insurance_copy.charges.value_counts())
print("-"*40)
```

```
sex: male    676
female   662
Name: sex, dtype: int64
```

```
smoker: no    1064
yes     274
Name: smoker, dtype: int64
```

```
age: 18    69
19    68
50    29
51    29
47    29
46    29
45    29
20    29
48    29
52    29
22    28
49    28
54    28
53    28
21    28
26    28
24    28
25    28
28    28
27    28
23    28
```

```
23    28
43    27
29    27
30    27
41    27
42    27
44    27
31    27
40    27
32    26
33    26
56    26
34    26
55    26
57    26
37    25
59    25
58    25
36    25
38    25
35    25
39    25
61    23
60    23
63    23
62    23
64    22
Name: age, dtype: int64
```

```
bmi: 32.300    13
28.310      9
30.495      8
30.875      8
31.350      8
..
```

```
46.200      1
23.800      1
44.770      1
32.120      1
30.970      1
Name: bmi, Length: 548, dtype: int64
-----
children: 0    574
1    324
2    240
3    157
4    25
5    18
Name: children, dtype: int64
-----
charges: 1639.56310    2
16884.92400    1
29330.98315    1
2221.56445    1
19798.05455    1
..
7345.08400    1
26109.32905    1
28287.89766    1
1149.39590    1
29141.36030    1
Name: charges, Length: 1337, dtype: int64
```

# Data Representation and Transformation with Python

[11]:

```
#Check for missing values in pandas dataframe using isnull() function
print('Data columns with null values:',insurance_copy.isnull().sum())
```

```
Data columns with null values: age      0
sex        0
bmi        0
children   0
smoker     0
region     0
charges    0
dtype: int64
```

[12]:

```
#Check for missing values in pandas dataframe using isna() function
df.isna().any()
```

```
[12]: age      False
       sex     False
       bmi     False
       children  False
       smoker  False
       region  False
       charges False
       dtype: bool
```

# Data Representation and Transformation with Python

```
[13]: #Check for duplicate values, It returns a boolean series which is True only for Unique elements  
df.duplicated()
```

```
[13]: 0      False  
1      False  
2      False  
3      False  
4      False  
...  
1333    False  
1334    False  
1335    False  
1336    False  
1337    False  
Length: 1338, dtype: bool
```

```
[14]: #Check how many rows have duplicate values  
df.duplicated().sum()
```

```
[14]: 1
```

```
[15]: # Extract duplicate rows  
df.loc[df.duplicated(keep=False), :]
```

```
[15]:   age  sex  bmi  children  smoker    region  charges  
195  19  male  30.59       0     no  northwest  1639.5631  
581  19  male  30.59       0     no  northwest  1639.5631
```

```
[16]: # Remove duplicate row  
df.drop_duplicates(inplace=True)
```

# Data Representation and Transformation with Python

[17]:

```
#Prints information about the DataFrame.  
#The information contains the number of columns, column labels, column data types, memory usage, range index, and the number of cells in each co  
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
Int64Index: 1337 entries, 0 to 1337  
Data columns (total 7 columns):  
 #   Column   Non-Null Count  Dtype     
---  --      --      --      --  
 0   age      1337 non-null   int64    
 1   sex      1337 non-null   object   
 2   bmi      1337 non-null   float64  
 3   children  1337 non-null   int64    
 4   smoker    1337 non-null   object   
 5   region    1337 non-null   object   
 6   charges   1337 non-null   float64  
dtypes: float64(2), int64(2), object(3)  
memory usage: 83.6+ KB
```

+ Code

+ Markdown

[18]:

```
#Transform sex=males. It returns a boolean series and set it as 'True', else 'False'  
df['sex'] == 'male'
```

1-Hot Encoding

```
0      False  
1      True  
2      True  
3      True  
4      True  
...  
1333     True  
1334     False  
1335     False  
1336     False  
1337     False  
Name: sex, Length: 1337, dtype: bool
```

# Data Representation and Transformation with Python

```
[19]:  
#Transform sex=male. It returns a value as '1' if True, else '0' if False  
(df['sex'] == 'male').astype(int)
```

## 1-Hot Encoding

```
[19]: 0      0  
1      1  
2      1  
3      1  
4      1  
..  
1333   1  
1334   0  
1335   0  
1336   0  
1337   0  
Name: sex, Length: 1337, dtype: int64
```

```
[20]: df['sex'] = (df['sex'] == 'male').astype(int)
```

```
[21]:  
#Transform smoker=yes. It returns a boolean series and set it as 'True', else 'False'  
df['smoker'] == 'yes'
```

```
[21]: 0      True  
1     False  
2     False  
3     False  
4     False  
..  
1333  False  
1334  False  
1335  False  
1336  False  
1337  True  
Name: smoker, Length: 1337, dtype: bool
```

# Data Representation and Transformation with Python

[22]:

```
#Transform smoker=yes. It returns a value as '1' if True, else '0' if False  
(df['smoker'] == 'yes').astype(int)
```

```
[22]: 0      1  
1      0  
2      0  
3      0  
4      0  
..  
1333    0  
1334    0  
1335    0  
1336    0  
1337    1  
Name: smoker, Length: 1337, dtype: int64
```

+ Code

+ Markdown

[23]:

```
df['smoker'] = (df['smoker'] == 'yes').astype(int)
```

[24]:

```
#Returns the first 5 rows of the dataframe  
df.head()
```

[24]:

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	1	southwest	16884.92400
1	18	1	33.770	1	0	southeast	1725.55230
2	28	1	33.000	3	0	southeast	4449.46200
3	33	1	22.705	0	0	northwest	21984.47061
4	32	1	28.880	0	0	northwest	3866.85520

[25]:

```
#Returns the last 5 rows of the dataframe  
df.tail()
```

[25]:

	age	sex	bmi	children	smoker	region	charges
1333	50	1	30.97	3	0	northwest	10600.5483
1334	18	0	31.92	0	0	northeast	2205.9808
1335	18	0	36.85	0	0	southeast	1629.8335
1336	21	0	25.80	0	0	southwest	2007.9450
1337	61	0	29.07	0	1	northwest	29141.3603

## 1-Hot Encoding

# Data Representation and Transformation with Python

001

```
[26]: #To find which User paid lowest premium  
df.charges.idxmin()
```

```
[26]: 940
```

```
[27]: #Retrieve specific rows from Dataframe  
df.iloc[940]
```

```
[27]: age      50  
sex      0  
bmi     46.09  
children    1  
smoker     0  
region    southeast  
charges   9549.5651  
Name: 941, dtype: object
```

+ Code

+ Markdown

```
[28]: #Returns index of first occurrence of maximum over requested axis.  
df.charges.idxmax()
```

```
[28]: 543
```

# Data Representation and Transformation with Python

```
[29]:  
    #Retrieve specific rows from Dataframe  
    df.iloc[543]
```

```
[29]: age      54  
       sex      0  
       bmi     47.41  
       children      0  
       smoker      1  
       region   southeast  
       charges  63770.42801  
Name: 543, dtype: object
```

```
[30]:  
    #Retrieve specific rows from Dataframe  
    df.iloc[581]
```

```
[30]: age      39  
       sex      1  
       bmi     45.43  
       children      2  
       smoker      0  
       region   southeast  
       charges  6356.2707  
Name: 582, dtype: object
```

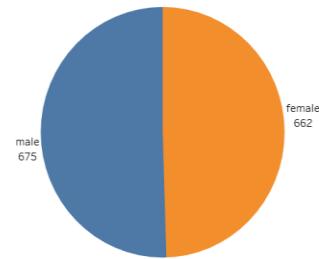
```
[31]:  
    #Retrieve specific rows from Dataframe  
    df.iloc[195]
```

```
[31]: age      19  
       sex      1  
       bmi     30.59  
       children      0  
       smoker      0  
       region   northwest  
       charges  1639.5631  
Name: 195, dtype: object
```

# Data Visualization with Tableau

## – What have we discovered from the data?

Total by sex



Total count of 1337 customers. sex breakdown is quite even.

- 51% Male
- 49% Female

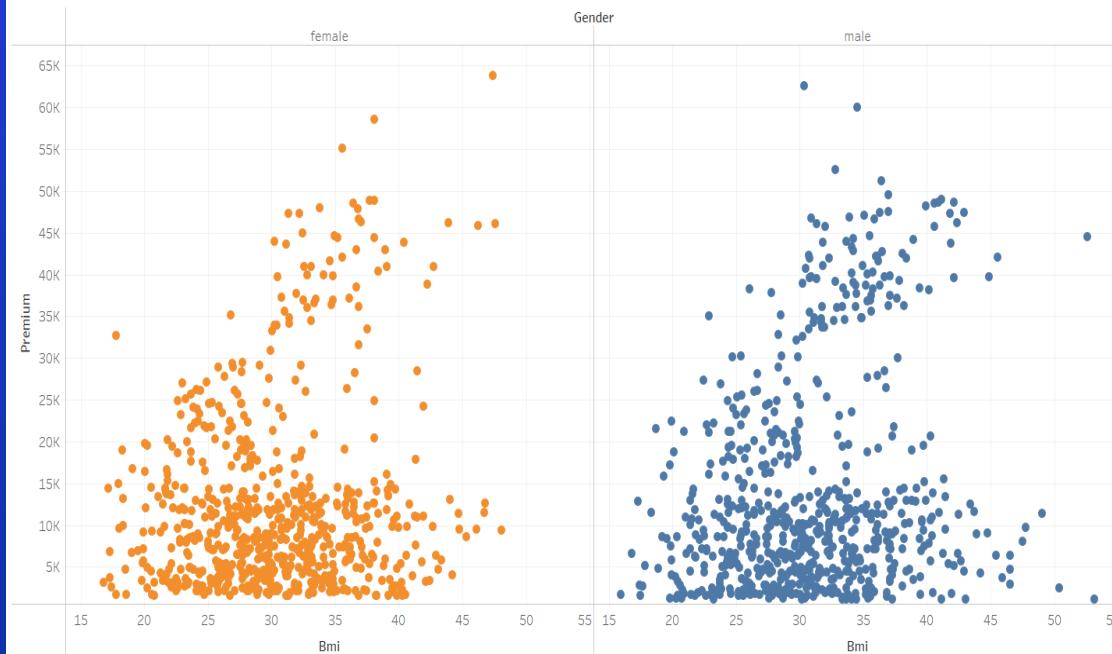
charges vs age (sex)



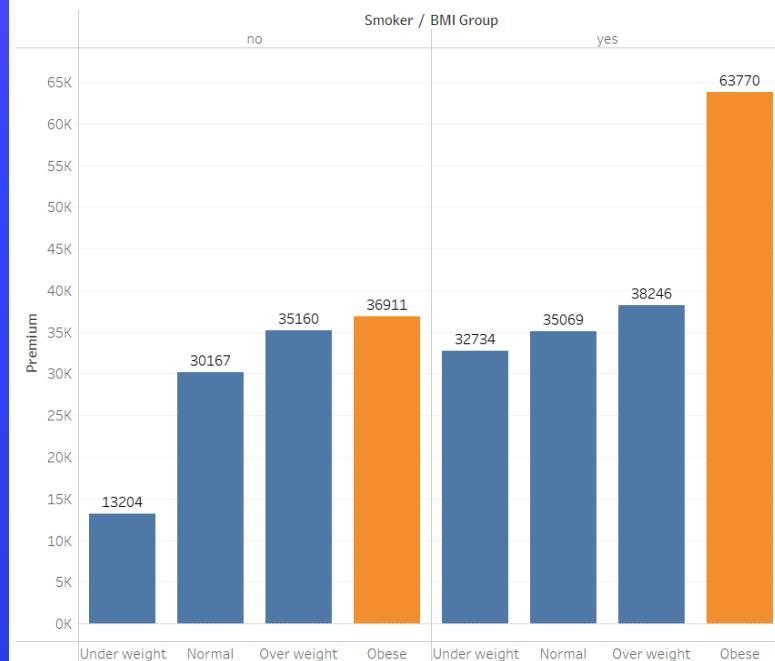
Positive Relationship between charges and age. Charges starts to go up as customer ages, regardless of sex. However, males do pay higher as they age.

# Data Visualization with Tableau - So, age does makes a difference. What about BMI?

charges vs bmi (sex)

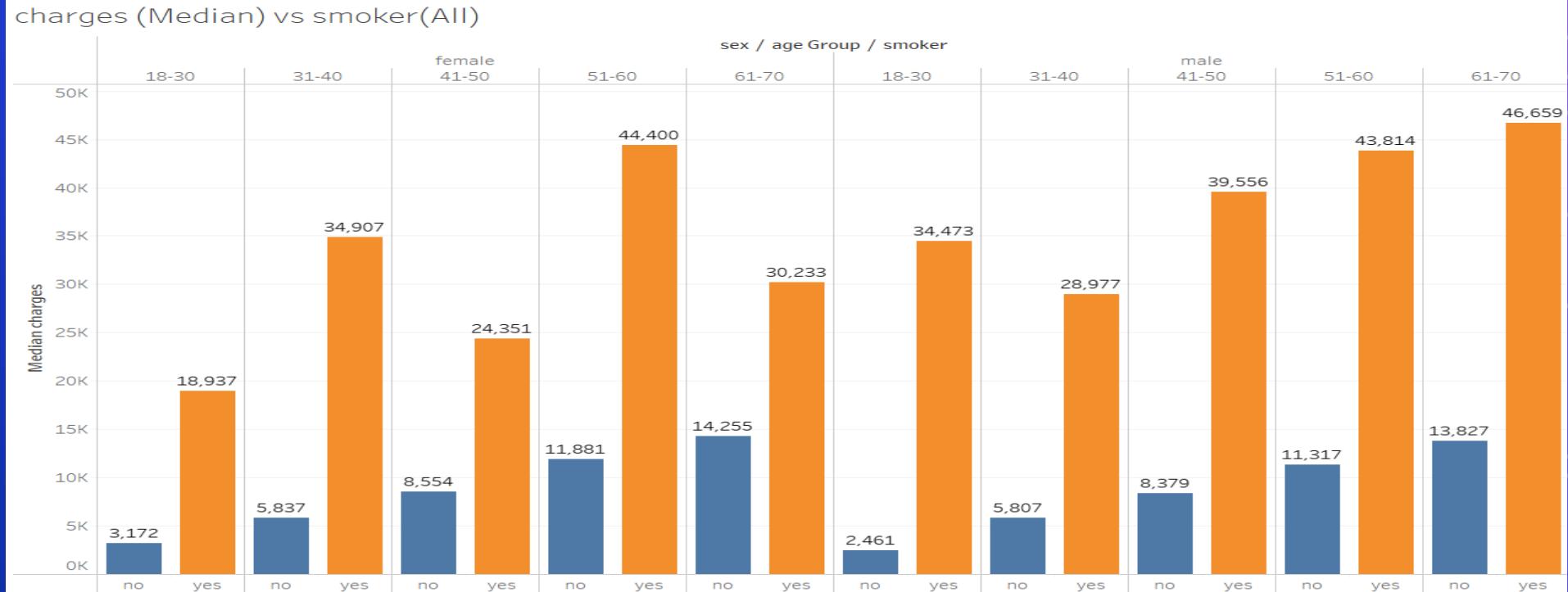


charges vs bmi/smoker



Charges paid by consumers with high BMI >30, tend to pay higher. But it is not always the truth based on the plot shown. But if BMI is high and smoking, charges paid is 74.3% higher then BMI high and non-smoker

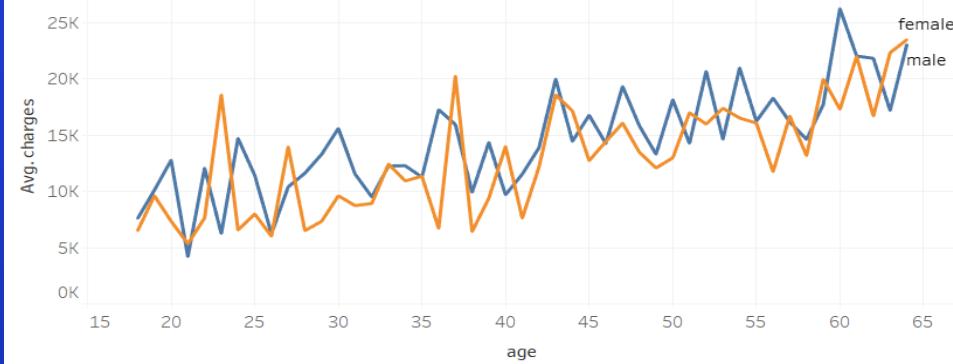
# Data Visualization with Tableau - Since, age and BMI make a difference. What about smokers?



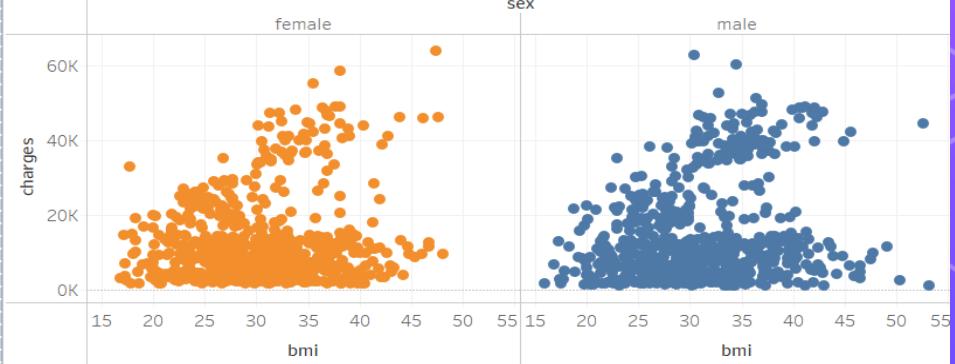
- Smokers across all ages and sex, pay higher than non-smokers.
- Male smoker pay over \$32k more compared to non-smoker for age group 18-30, 61-70

# Dashboard from Tableau

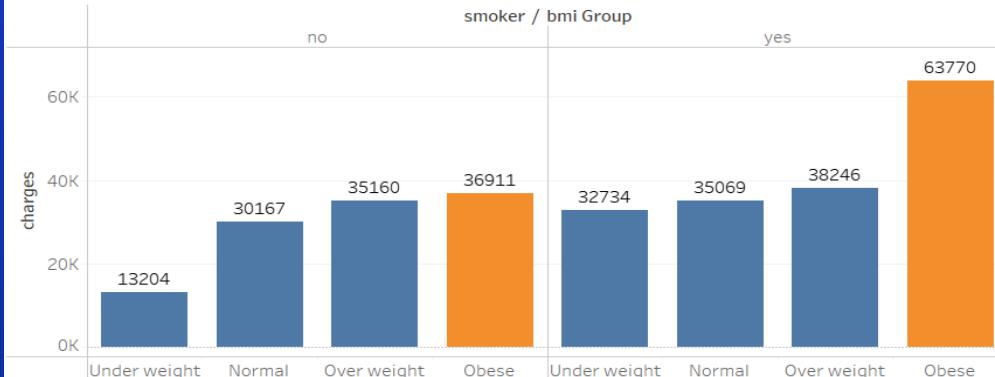
charges vs age (sex)



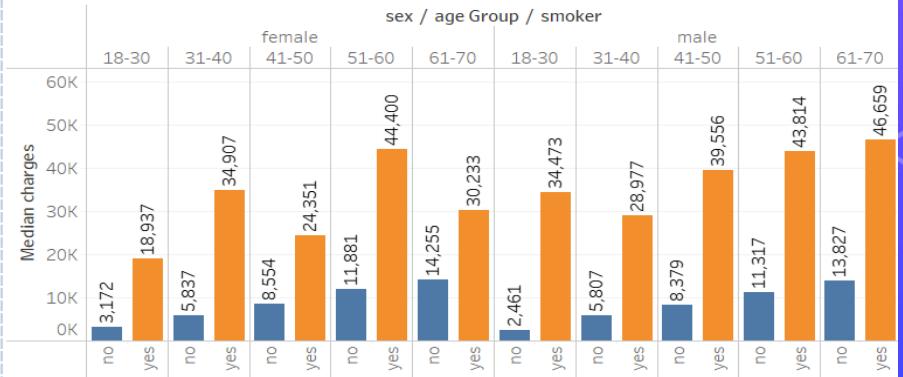
charges vs bmi (sex)



charges vs bmi/smoker



charges (Median) vs smoker(All)



# 4. Diagnostic Analytics



# Attribute Insights

With these findings, what can we conclude?

As evident by the analysis thus far, we can conclude that there is a strong relationship between charges charged and the various key variables. This has been explained by the health risk associated with high BMI readings and smoking habit. Thus charges are charged higher to consumers with such traits.

Key Variables	Affect Charges
Age ↑	✓
BMI ↑	✓
Smoker ✓	✓

**Conclusion:**  
**Anyone with BMI and smoke at the same time, will have to be prepared for high charges when buying a policy or renewal, this aligns with our hypothesis**

# 5. Predictive Analytics



# Predictive Analytics - What Happens next?

- Using descriptive data accumulated over time to find explanatory variables, we are able to build statistical models that can be applied to other data points for which we're to predict the outcome.
- Our Team **identified an issue** where Users are not aware which factors has the most weightage affecting the calculation of their health insurance charges
- **Discovered** that the issue is causing Users not knowing what action they can take to have greater financial savings, i.e. pay lesser charges

We need to:

- **Predict** the expected health insurance charges that the Users need to pay given their biometrics demographic and make recommendations accordingly.

Our team decided to use AutoAI for this section.

# What is Auto AI?

Automated machine learning (AutoML) is the process of automating the manual tasks that data scientists must complete as they build and train machine learning models (ML models).

These tasks include

- feature engineering and selection,
- choosing the type of machine learning algorithm;
- building an analytical model based on the algorithm;
- hyperparameter optimization,
- training the model on tested data sets and
- running the model to generate scores and findings.



**AutoAI is a variation of AutoML. It extends the automation of model building to the entire AI lifecycle.**

**Like AutoML, AutoAI applies intelligent automation to the steps of building predictive machine learning models.**

# Why we choose to use Auto AI?



Build Models Faster



Find signal from noise



Jump The skills gap



Rank and explore models

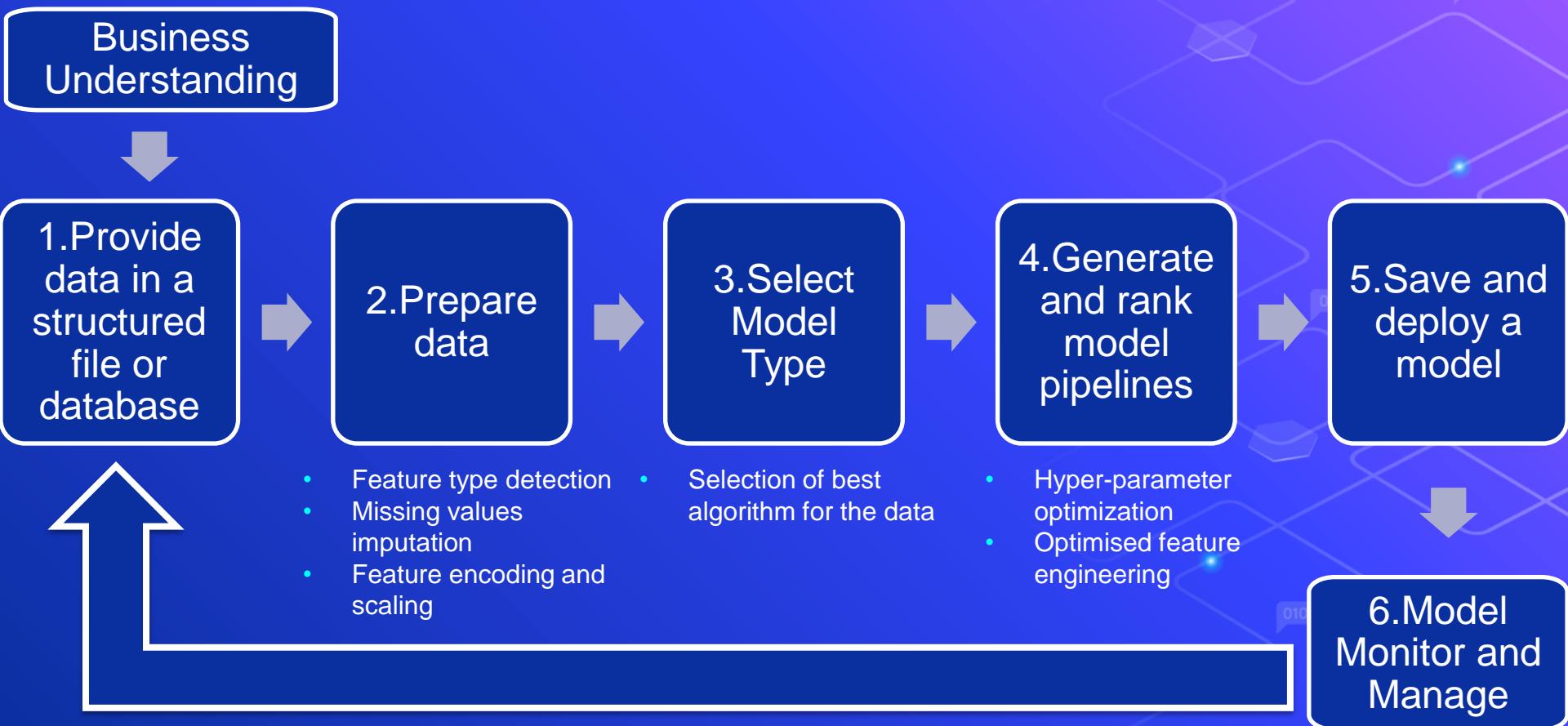


Discover more use cases



Ready, set, deploy

# Auto-AI process



# Train Data Models with Auto AI

Experiment summary

Relationship map ⓘ  
Prediction column: charges

Pipeline comparison

★ Rank by: Root mean squared error (RMSE) (...) | Cross validation score ⚙

AutoAI will use 90% of the training data to generate AutoAI model pipelines and use cross-validation scores to rank model pipelines

insurance.csv

90% TRAINING DATA  
3 folds

10% HOLDOUT DATA

After the AutoAI training is completed, the Holdout Data is used for model evaluation and computation for the performance information.

**Root Mean Squared Error (RMSE)** is the square root of the mean of the square of all of the error.

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (S_i - O_i)^2}$$

where  $O_i$  are the observations,  $S_i$  predicted values of a variable, and  $n$  the number of observations available for analysis.

**RMSE is a good measure of accuracy, but only to compare prediction errors of different models or model configurations for a particular variable and not between variables, as it is scale-dependent.**

The use of RMSE is very common, and it is considered an excellent general-purpose error metric for numerical predictions.

Pipeline leaderboard ▾

49

# Train Data Models with Auto AI

Experiment summary Pipeline comparison ★ Rank by: Root mean squared error (RMSE) ... | Cross validation score ⚖ 001

Relationship map ⓘ  
Prediction column: charges

SELECTING ALGORITHMS

10% of training data  
insurance.csv

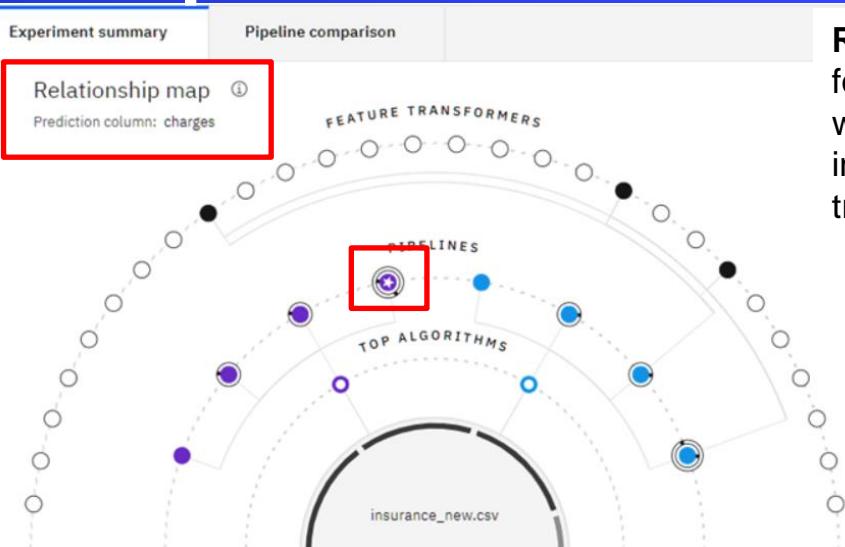
Progress map

**Model Selection – AutoAI**  
will only keep top 2 performing algorithms and discards the remaining underperforming algorithms

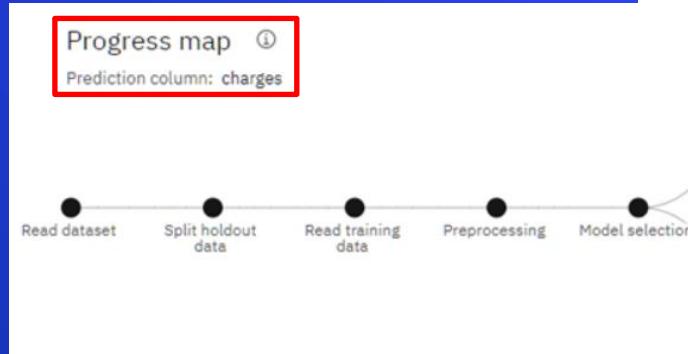
Model selection  
INSURANCE.CSV  
  
Selecting algorithms for pipeline generation using 10% of training data. Discarding underperforming algorithms and keeping the top 2 algorithms.  
  
Time elapsed: 83 seconds

[View log](#) [Save code](#)

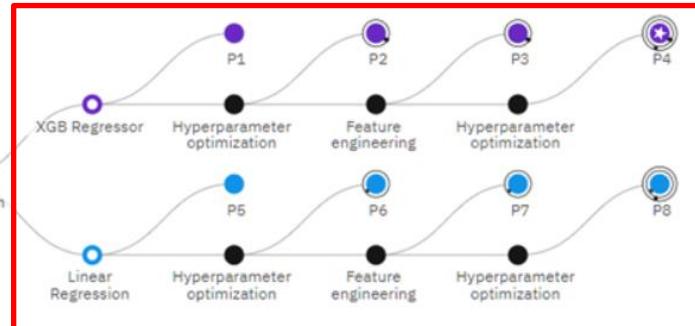
# Pipeline Transformations



**Relationship map view** is useful for specific pipeline view where we can hover over a node in the infographic to view the feature transformations for that pipeline



**Progress Map view** is useful to show the overview of the entire AutoAI process, which are the top 2 algorithms kept; and if any Hyperparameter optimization or feature engineering was performed for the pipeline.



# Pipeline Leaderboard

Pipeline leaderboard ▾

**XGBoost** is an efficient implementation of gradient boosting that can be used for regression predictive modelling.

FE - Feature engineering attempts to convert feature values into usable metrics.

Rank	↑	Name	Model	Score	Enhancements	Time
1		Pipeline 4	XGB Regressor	4598.614	HPO-1 FE HPO-2	00:00:40
2		Pipeline 2	XGB Regressor	4640.911	HPO-1	00:00:14
3	Pi	For each fold and algorithm type, AutoAI creates two pipelines that use HPO to optimize the algorithm type and you can see Pipeline 4 using both HPO-1 & HPO-2 while Pipeline 2 is only using HPO-1		What RBfOpt does is that it fits a radial basis function mode to accelerate the discovery of hyper-parameter configurations that maximize the objective function of the machine learning problem at hand.		
4	Pi					
5	Pi					
6	Pi					

**Hyper-parameter optimization (HPO)** uses a model-based, derivative-free global search algorithm, called RBfOpt, which is tailored for the costly machine learning model training and scoring evaluations required by HPO

# Metric Chart

Projects / DSTeam2\_PredictInsuranceAmt / PredictInsuranceCharges

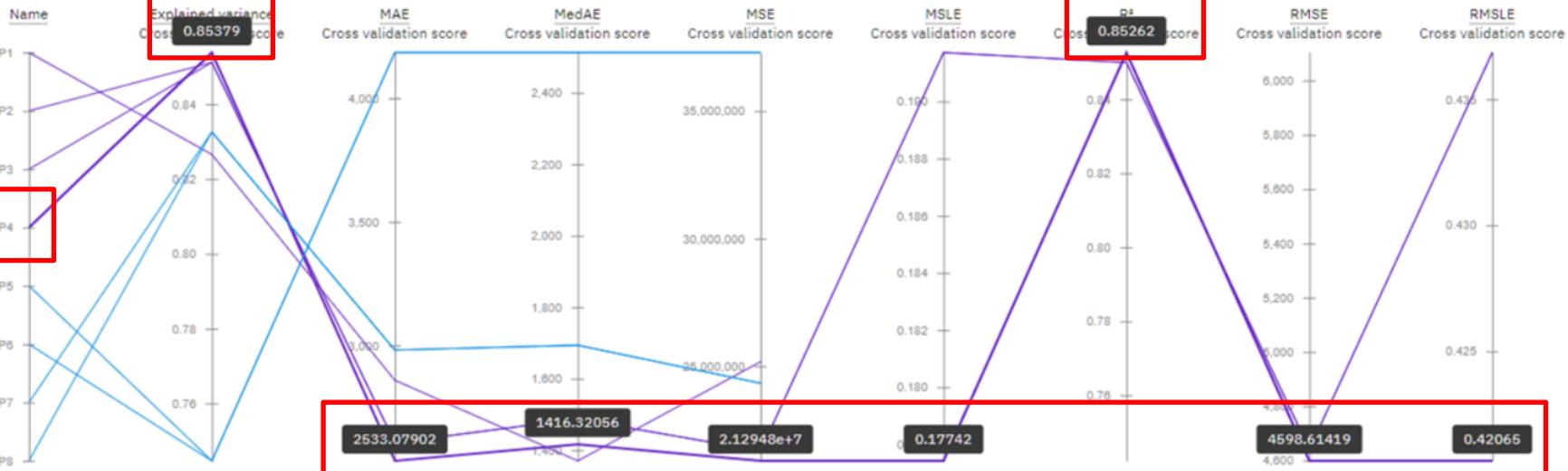
Experiment summary

Pipeline comparison

★ Rank by: Root mean squared error (RMSE) (... | Cross validation score

Metric chart ⓘ

Prediction column: charges



# Pipeline Details – Feature Summary

Pipeline details

Pipeline 4

Rank: 1 | RMSE (Optimized): 4138.957 (Holdout) | Algorithm: XGB Regressor | Enhancements: HPO-1, FE, +1

Save as

Model viewer

Model information

Feature summary

Evaluation

Model evaluation

Feature summary

Original features

Search feature or transformer names

High correlation

Feature summary

Feature name	Transformation	Feature importance								
smoker	None	100.00%								
bmi	None	4.00%								
age	None	2.00%								
children	<table border="1"><thead><tr><th>Key Variables</th><th>Affect Charges</th></tr></thead><tbody><tr><td>Age</td><td>✓</td></tr><tr><td>BMI</td><td>✓</td></tr><tr><td>Smoker</td><td>✓</td></tr></tbody></table>	Key Variables	Affect Charges	Age	✓	BMI	✓	Smoker	✓	1.00%
Key Variables	Affect Charges									
Age	✓									
BMI	✓									
Smoker	✓									
sex		0.00%								
region		0.00%								

54

# Pipeline Details – Model Evaluation

001

Pipeline details

Pipeline 4

Rank RMSE (Optimized) Algorithm Enhancements

1 4138.957 (Holdout) XGB Regressor HPO-1 FE +1

Save as

Model viewer

Model information

Feature summary

Evaluation

Model evaluation

### Model evaluation ①

#### Model evaluation measure

Measures	Holdout score	Cross validation score
Root mean squared error	4138.957	4598.614
R squared	0.895	0.853
Explained variance	0.895	0.854
Mean squared error	17130968.000	21294802.667
Mean squared log error	0.202	0.177
Mean absolute error	2347.269	2533.079
Median absolute error	1533.956	1416.321
Root mean squared log error	0.449	0.421

# 6. Prescriptive Analytics



# Prescriptive Analytics – What should you do?

- Prescriptive analytics take the inputs from prediction and combined with rules and constraint-based optimization, this enables better decisions about what to do.
- If we want to change the way we do business, first we need to be able to predict future outcomes and variables that may impact those outcomes.
- Now that we've identified what's going to happen, we need to determine what to do, in order to overcome operational obstacles and achieve continuous improvement

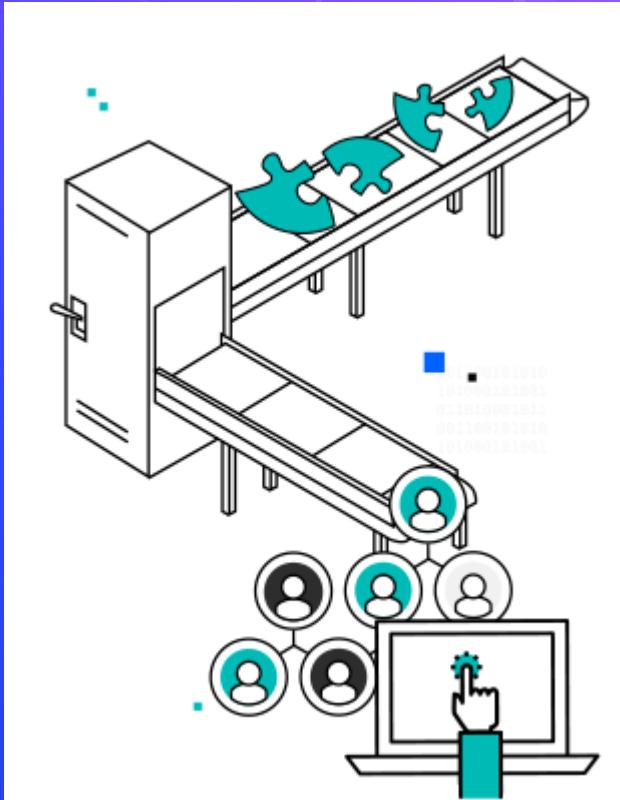
# Before Model Deployment

001

Questions to answer:

- What are my deployment requirements?
- How will I evaluate the model's performance in production?
- Have I minimized the trap of overfitting my model?
- How frequently do I plan on re-training my model?
- What are the data preprocessing needs?
- Will the format of the production input data differ drastically from the model training data?
- Will it come in batches or as a stream?
- Do I need to run the model offline?

Models are normally deployed in a limited way until their performance has been fully evaluated. Deployment may be as simple as generating a report with recommendations, or as involved as embedding the model in a complex workflow and scoring process managed by a custom application.



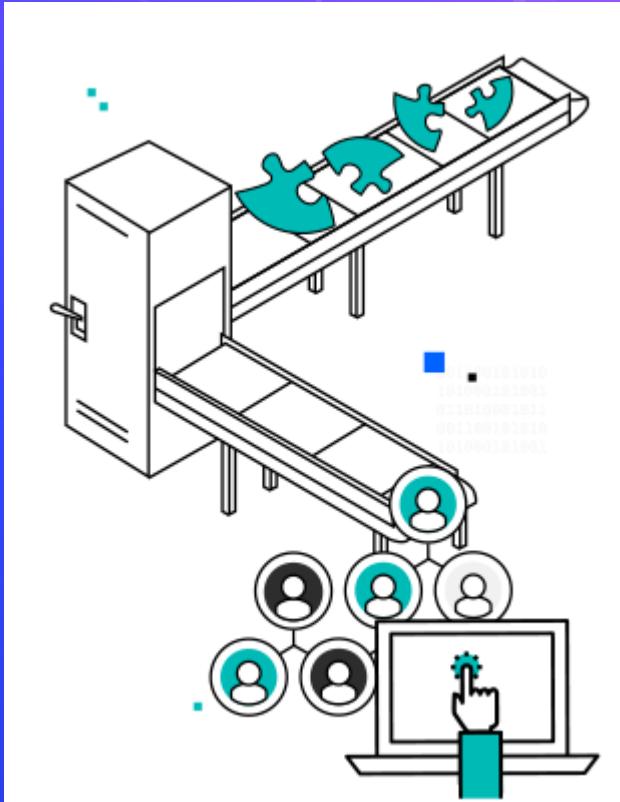
# Model Deployment

Once a satisfactory model has been developed and is approved by the business sponsors, we are now ready to deploy it into the production environment or a comparable test environment.

While deploying a model into an operational business process is usually an IT function, the process is not as simple as simply handing the model off to the IT team.

Testing the model prior to deployment is necessary to identify any dependencies in the production environment.

We also need to ensure models are able to receive correct production data and send the scores to the right place, and that the system must be set up for monitoring and scalability post deployment.



# Model Deployment

Projects / DSTeam2\_PredictInsuranceAmt / PredictInsuranceCharges - P4 X...



## PredictInsuranceCharges - P4 XGB Regressor

PredictInsuranceCharges - P4  
XGB Regressor

Promote to deployment space

### Input Schema

Input	
Column	Type
age	"integer"
bmi	"double"
children	"integer"
region	"other"
sex	"integer"
smoker	"integer"



PredictInsuranceCharges - P4  
XGB Regressor

Last modified at Apr 20, 2022 7:28 AM

Description  
No description provided.

Created  
Apr 20, 2022 7:28 AM

Type  
wml-hybrid\_0.1

Model ID  
025b6d76-419c-4e03-93f3-a2...

Software specification  
hybrid\_0.1

Hybrid pipeline software specifications  
autoai-kb\_rt22.1-py3.9

Tags  
Add tags to make assets easier to find.

# Model Deployment

## Promote to space

Use a deployment space to organize supporting resources such as input data and environments; deploy models or functions to generate predictions or solutions; and view or edit deployment details.

## Create a deployment space

Use a space to collect assets in one place to create, run, and manage deployments

### Define space details

#### Name

PredictInsuranceCharges

#### Description (Optional)

Deployment space description

### Select storage service ⓘ

The space is ready

Close this notification to resume your work. Click **Deployments** in the navigation pane to view and access the new space.

✓ Step 1 of 1. Creating deployment space.

**Close**

Cancel

Create

### Deployment space tags (optional) ⓘ

Add a tag

001

# Model Deployment

001

## Promote to space

Use a deployment space to organize supporting resources such as input data and environments; deploy models or functions to generate predictions or solutions; and view or edit deployment details.

Target space

PredictInsuranceCharges

Go to the model in the space after promoting it

Selected assets (1)

Asset name	Format
PredictInsuranceCharges - P4 XGB Regressor	Model

Select version

*i* Promoting a version of an asset to a space creates a new asset in the space, with a new asset ID.

Current

Description (optional)

Description of assets

Cancel

Promote

# Model Deployment

001

## Create a deployment

### Associated asset

PredictInsuranceCharges - P4 XGB Regressor

### Deployment type

#### Online

Run the model on data in real-time, as data is received by a web service.

#### Batch

Run the model against data as a batch process.

### Name

PredictInsuranceCharges

### Serving name ①

Deployment serving name

### Description

Cancel

Create

# Model Deployment

The screenshot shows the Model Deployment interface for the space "PredictInsuranceCharges". The top navigation bar includes "Deployments /", "Add to space", and a notification icon (001). The main content area has tabs for "Overview", "Assets", "Deployments" (selected), "Jobs", and "Manage".

**Deployments:** Shows 1 Deployed and 0 Failed.

**Space activity:** Displays two notifications:

- Online deployment ready:** The online deployment "PredictInsuranceCharges" in space "PredictInsuranceCharges" is ready to accept requests. (Today at 07:32 AM)
- Online deployment created:** You created online deployment "PredictInsuranceCharges" in space "PredictInsuranceCharges". You must wait for the deployment to enter ready state before you can accept requests. (Today at 07:31 AM)

**Upload Area:** A dashed box on the right allows users to "Drop files here or browse for files to upload." It also instructs users to "Stay on the page until upload completes. Incomplete uploads are cancelled."



# Model Deployment

PredictInsuranceCharges Deployed Online

API reference Test

Direct link

Endpoint Bearer <token> ⓘ

<https://us-south.ml.cloud.ibm.com/ml/v4/deployments/abfdf0b9-238e-42ce-ae7b-b8486e513b1e/predict> ⚙️ IAM

Code snippets

cURL Java JavaScript Python Scala

```
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "<your API key>"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey": API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
```

# Model Deployment

Deployments / PredictInsuranceCharges / PredictInsuranceCharges - P4 X...



## PredictInsuranceCharges

Deployed Online

API reference

Test

Enter input data

Input list (1)

Result

age

Integer

sex

Integer

bmi

[ 60, 0, 32.45, null, 1, null ]



```
0 i
1   "predictions": [
2     i
3       "fields": [
4         "prediction"
5       ],
6       "values": [
7         [
8           45550.74609375
9         ]
10      ]
11    ]
12  ]
13 }
```

	A	B	C	D	E	F	G
1	age	sex	bmi	children	smoker	region	charges
22	60	0	36.005	0	0	northeast	13228.85
435	60	0	30.5	0	0	southwest	12638.2
847	60	0	32.45	0	1	southeast	45008.96
906	60	0	35.1	0	0	southwest	12644.59

Same Age, Sex, BMI,  
Smoker status  
Predict: 45,550.74  
Actual: 45,008.96  
Variance: +541.785

# Model Comparison using Python

001

```
# Import Libraries and DataSet
import numpy as np
import pandas as pd

dataset = pd.read_csv('../input/insurance-new2/insurance_new2.csv')
```

+ Code + Markdown

```
# Display dataset
print (dataset.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 6 columns):
 #   Column   Non-Null Count  Dtype  
 --- 
 0   age      1338 non-null   int64  
 1   sex      1338 non-null   int64  
 2   bmi      1338 non-null   float64
 3   children 1338 non-null   int64  
 4   smoker    1338 non-null   int64  
 5   charges   1338 non-null   float64
dtypes: float64(2), int64(4)
memory usage: 62.8 KB
None
```

+ Code + Markdown

```
# Create Dataframe
df = dataset

# Specify which columns for input variables
X = dataset.iloc[:, :-5]

# Specify which column is the target or predict column
y = dataset.iloc[:, -5]
```

```
# Training model using linear regression
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X.values, y)
```

```
# Testing Model for a Age=60, Sex=0, BMI=32.45, Smoker=1
# Predict output for Charges to be $45,008.96
print(regressor.predict([[60, 0, 32.45, 0, 1]]))
```

[37695.74856522]

```
# Training model using XGBRegressor (Similar Algorithm as Auto AI)
from xgboost import XGBRegressor
my_model = XGBRegressor()
my_model.fit(X.values, y)
```

```
# Testing Model for a Age=60, Sex=0, BMI=32.45, Smoker=1
# Predict output for Charges to be $45,008.96
print(my_model.predict([[60, 0, 32.45, 0, 1]]))
```

[45044.88]

Actual: \$45,008.96

Python

Linear Regression: \$37,695.75

XGB Regressor: \$45,044.88\*\*

Auto AI

XGB Regressor :\$45,550.74

Result

```
0: {
  "predictions": [
    {
      "fields": [
        "prediction"
      ],
      "values": [
        45550.74609376
      ]
    }
  ]
}
```

Conclusion:

Both machine learning methods provides similar results if similar algorithms are used. Auto AI provides the convenience to automate the testing of 8 algorithms and ranking the best performing algorithm pipeline which otherwise, we will need to do manually like this example. However, it will be useful if Auto AI can display all 8 algorithms so that we are aware which 6 algorithms are discarded from the entire list of 11 regression algorithms.

# Model Deployment

Deployments / PredictInsuranceCharges / PredictInsuranceCharges - P4 X... /

PredictInsuranceCharges Deployed Online

API reference Test

Enter input data

integer	bmi	Double	children	Integer	smoker
---------	-----	--------	----------	---------	--------

Input list (1)

```
[ 20, 1, 27, null, 1, null ]
```

Result

```
0 {  
1   "predictions": [  
2     {  
3       "fields": [  
4         "prediction"  
5       ],  
6       "values": [  
7         [  
8           17966.62109375  
9         ]  
10      ]  
11    }  
12  ]  
13 }
```

Same Age, Sex, Smoker status, Slight different in BMI  
Predict: 17,966.62  
Actual: 16,232.85  
Variance: +1,733.77

A	B	C	D	E	F	G	H
age	sex	bmi	children	smoker	region	charges	
20	1	27.93	0	0	northeast	1967.023	
20	1	27.3	0	1	southwest	16232.85	

# Model Deployment

PredictInsuranceCharges

Deployed Online

Interactive Recommendation:

Assuming the User is a 48years male smoker with BMI 32.3, besides displaying the current prediction of \$42,339.88 at age 48.

age	sex	bmi	smoker	charges	Predict	Variance
48	1	32.3	0	8,765.25	9,804.94	1,039.69
50	1	32.3	1	41,919.10	42,749.02	829.92
50	1	32.3	0	9,630.40	10,010.84	380.44

An AI recommendation will also display to the user that while maintaining his current BMI; if he maintains his current smoking habit, the charges will increase slightly in 2 years time to \$42,749.02.

However, he could have at least \$32,000 in cost savings if he decides to give up smoking at age 50 and only pay \$10,010.84. Thus, encouraging the user to make a healthy change in his lifestyle NOW to enjoy greater financial benefits later.



- [ 50, 1, 32.3, null, 1, null ]
- [ 50, 1, 32.3, null, 0, null ]
- [ 48, 1, 32.3, null, 1, null ]
- [ 48, 1, 32.3, null, 0, null ]

```
0 { "predictions": [  
1   {  
2     "fields": [  
3       "prediction"  
4     ],  
5     "values": [  
6       [ 42749.01953125  
7     ],  
8     [ 10010.841796875  
9     ],  
10    [ 42399.8828125  
11    ],  
12    [ 9804.947265625  
13    ]  
14  ]  
15 }  
16 ]  
17 }  
18 010
```

Similar, if the user is currently a non-smoker, we will display in the UI a recommendation for user to continue his good non-smoker habit to avoid the need to pay additional \$32,000 charges in 2 years time.

# Environment Feedback (Reality Check)

- Collecting results from the implemented model allows organizations to get feedback on the model's performance.
- **Analysing** the feedback enables the data scientist to **refine**, increase **accuracy** and its **usefulness**. (yields substantial additional benefits if it is undertaken as part of the overall process)
- Through feedback, refinement and redeployment, a model should continually **adapt to conditions**, allowing both the model and the work behind it to provide **value to the organization**.
- This methodology shows the **iterative nature** of the problem-solving process. Meaning we have to build, refine and improve the project through continuous testing and revise until we reach to a satisfactory level.

# Environment Feedback (Reality Check)

- We need to avoid the most common problem in machine learning and data science – Overfitting.
- **Overfitting** refer to when a model that works well on training data but performs badly on new data, i.e. our model does not generalize well from our training data to unseen data.

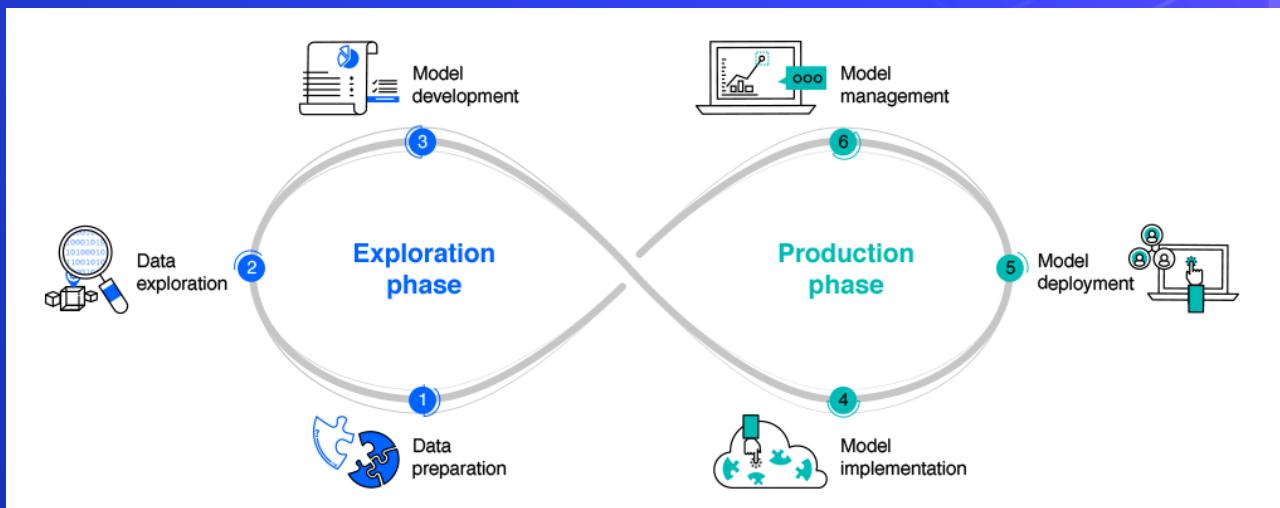
How to prevent Overfitting?

- Through **Model Management**

# Environment Feedback (Reality Check)

## Continuous Learning

- Using **continuous learning** paradigm, the data scientist can easily re-deploy and re-train the model pipeline using new data that may result in a better aligned prediction.



# Environment Feedback (Reality Check)

- **Model management** must be a continuous process to ensure optimal performance over time. i.e. knowing exactly where each model is in the lifecycle, how old the model is, who developed it, and who is using it for which application.
- **Model version control**, such as event logging and change tracking to understand how the model evolves over time, is another critical requirement.
- Data science must also be concerned with **model decay**, and **continuously gather metrics** to determine when a model should be refreshed or replaced.

# 7. Summary and Reflection



# Constraints and Challenges

Constraints	Challenges
Need to create multiple trial accounts due to limited available CU in AutoAI.	Short project timeline from choosing what data should be to analysed to creating a working model.
Multiple tools needed to be explored, in order to choose the right tool for the project. Group members need to learn and compare tools.	Not everyone is proficient in Python thus making it challenging for some of the group members.
Not easy to locate more similar dataset to further test the model (i.e. unseen data)	Takes time to choose the right visualization method/ tools to present the data.
Unable to open new IBM cloud Account due to failing IBM verification and old account being deactivated	Need to learn multiple platforms e.g. Python coding, Tableau, AutoAI and data refinery within a short time frame is challenging

# Future Enhancement

- Able to compare premium charges across different providers
- Able to suggest or show the benefit of staying with current provider or sign a new with another (in dollar terms)
- Able to pinpoint which criteria / underlying conditions causes the higher premium after generating a quote.
- Able to input additional parameters that could affect the insurance charges.  
Currently, user only given limited option to input personal data and underlying health conditions are not factored in and there is always a deviation from expected premium.
- Able to link to other apps that can help users monitor their health and lifestyle to that they can potentially keep their insurance premium to a minimum.

# Conclusion

- Data Science isn't just about frameworks, data sources and pipelines — it's also about people. So, having a team with diverse knowledge is vital to a successful project
- Following Data Science methodology steps helps us to handle and produce insights using the data in our hands to generate meaningful AI solutions.
- Practicing the Data Science lifecycle from data exploration to deployment and model management allows us to understand and utilize available data into affective business model faster and more accurately.
- Exploring various visualization tools that are available, not only Watson Studio, but also other platforms such as Tableau, Matplotlib, etc. is important as different tools have different pros and cons.

# The Journey Takeaways

- There is no 1 size fit all tool but it gives us opportunity to learn multiple tools.
- Using the right tool enables us to create the right model that reinforce our hypothesis.
- Understanding the data gives us the opportunity to create new business models.
- Having seen the power of diverse team brings different perspectives, skills and talents to this group project.
- Taking advantage of the strengths and weaknesses of our team mates allows us to be effective in creating the final outcome
- Every member of the team should learn basic coding to improve efficiency and also to be dynamic in using different tools.
- Prepare multiple trial accounts before hand and test out the accounts to make sure there are no errors.

# The Journey Takeaways

- We should spend more time exploring available datasets with additional data to include more parameters to make the prediction more accurate.
- Team members should take time to test out the datasets from refinery stage to deploying of model so that not only everyone understand the project but also get to explore tools and learn the differences and apply them accordingly.



Projects / DSTeam2\_PredictInsuranceAmt

Overview

Assets

Jobs

Manage

Find assets

8 assets

All assets

## Asset types

&gt; Data

2

&gt; Flows

2

&gt; Experiments

2

&gt; Saved models

2

## All assets

Name	Last modified
PredictInsuranceCharges - P4 XGB Regressor Model	1 week ago TL Law (You)
PredictInsuranceCharges AutoAI experiment	1 week ago TL Law (You)
insurance_new.csv_flow Data Refinery flow	1 week ago TL Law (You)
insurance_new.csv CSV	1 week ago TL Law (You)
PredictInsuranceAmt - P8 XGB Regressor Model	1 week ago TL Law (You)
PredictInsuranceAmt AutoAI experiment	1 week ago TL Law (You)
insurance.csv_flow Data Refinery flow	1 week ago TL Law (You)
insurance.csv CSV	1 week ago TL Law (You)

Items per page: 20 ▾

1–8 of 8 items

001

[Search or jump to...](#)

[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

[JLaw-TL / IMVAI1901\\_Team2](#) [Public](#)

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

[main](#) [1 branch](#) [0 tags](#)

[Go to file](#) [Add file](#) [Code](#)

 [JLaw-TL](#) Update README.md b059f31 5 hours ago 7 commits

 [IMVAI1901-DS-Team2-Insurance-Fin...](#) Visualization using Tableau 2 days ago

 [README.md](#) Update README.md 5 hours ago

 [datarepresentation-transformation.ip...](#) Kaggle Notebook | DataRepresentation\_Transformation | Version 3 5 hours ago

 [ds-team2-predict-insurance-charges...](#) Kaggle Notebook | DS-Team2\_Predict\_Insurance\_charges\_RegressionMo... 2 days ago

[README.md](#)

DataScience project for IBM IMVAI 1901 Team 2.

Project Objective

Using Data analytics Lifecycle and following the Data Science Methodology, our objective is to accurately predict health insurance charges and provide recommendations to our Users on what they do to avoid paying a higher health insurance charges than other Users with similar biometrics to achieve greater financial savings.

[https://github.com/JLaw-TL/IMVAI1901\\_Team2](https://github.com/JLaw-TL/IMVAI1901_Team2)

# Thanks!

Special thanks to Suhaimi

