

Predicting Compressive Strength of Concrete using IBM Watson AutoAI Experiment

Introduction :

➤ Overview -

The project is about predicting compressive strength of concrete using IBM Watson Auto AI Experiment. This comes under the category of "Machine Learning". The main objective of the project is to predict the strength of the concrete.

➤ Purpose -

Concrete is a material used in construction that has great versatility and which is used across the globe. Concrete has several advantages, including good compressive strength, durability, workability, construction availability, and low cost. Determining accurate concrete strength is a major civil engineering problem. Test results of 28-day concrete cylinder represent the characteristic strength of the concrete that has been prepared and cast to form the concrete work. It is important to wait 28 days to ensure the quality control of the process, although it is very time consuming. Thereby, this model helps to predict the compressive strength of concrete from early age test results.

Literature Survey :

➤ Existing Problem -

Determining accurate concrete strength is a major civil engineering problem. It takes 28 days to know the characteristic strength of the concrete that has been prepared. So it is important to wait till 28 days to ensure the quality of the concrete. Therefore, it is very time consuming.

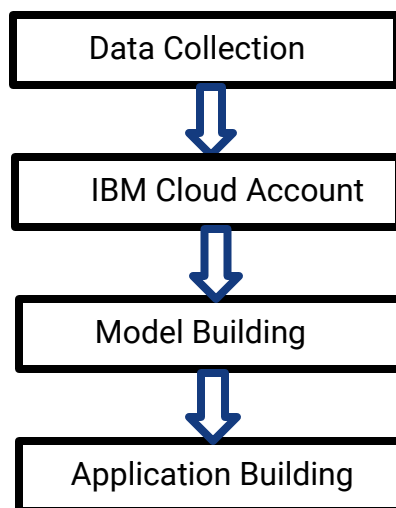
➤ Proposed Solution -

An ability to predict the compressive strength of concrete early allows constructors to quickly understand the concrete's probable weaknesses and make a decision to manage a destruction process or continue with construction. Further, to the benefit of both user (and purchaser) and producer, reliably and rapidly predicting the results of a 28-day test would benefit all stakeholders as opposed to waiting the full, conventional, 28 days. We are building a Machine Learning model to predict the compressive strength of concrete using IBM Watson AutoAI Machine Learning Service. The model is deployed on IBM cloud to get scoring end point which can be used as API in mobile app or web app building. We are developing a web application which is built using node red service. We make use of the scoring end point to give user input values to the deployed model. The model prediction is then showcased on User Interface.

Theoretical Analysis :

➤ Block Diagram -

Steps to be followed to build our application:



➤ Hardware/Software Designing :

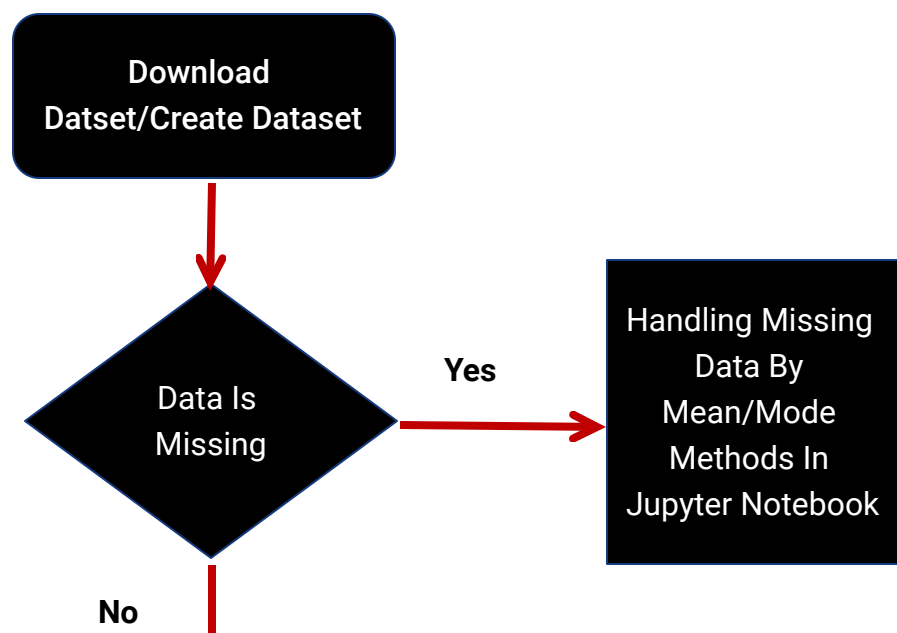
- Jupyter Notebook for data analysis.
- IBM Cloud
- Creating IBM Watson Studio AutoAI Experiment.
- Building user interface with Nodered.

Experimental Investigations :

Upon analysis of the dataset we understand that :

- As the dataset consists of multiple independent input attributes and an output attribute is to be predicted, it comes under supervised learning.
- As the output is continuous and not categorical, regression algorithm should be used, but as we are using AutoAI here, any regression is not applied externally.
- Since there are multiple columns, we have to analyse which columns have importance using various methods and extract the important ones.
- So from the visualization we can say that cement, water, superplasticizer and age are the important parameters for determining the strength of the concrete.

Flowchart :



```
graph TD; A[ ] --> B[IBM Cloud Registration]; B --> C[Login To IBM Cloud]; C --> D[Create Cloud Object Storage]; D --> E[Create Watson Studio Platform]; E --> F[Create Machine Learning Service]; F --> G[Create Project In Watson Studio]; G --> H[ ]
```

IBM Cloud Registration

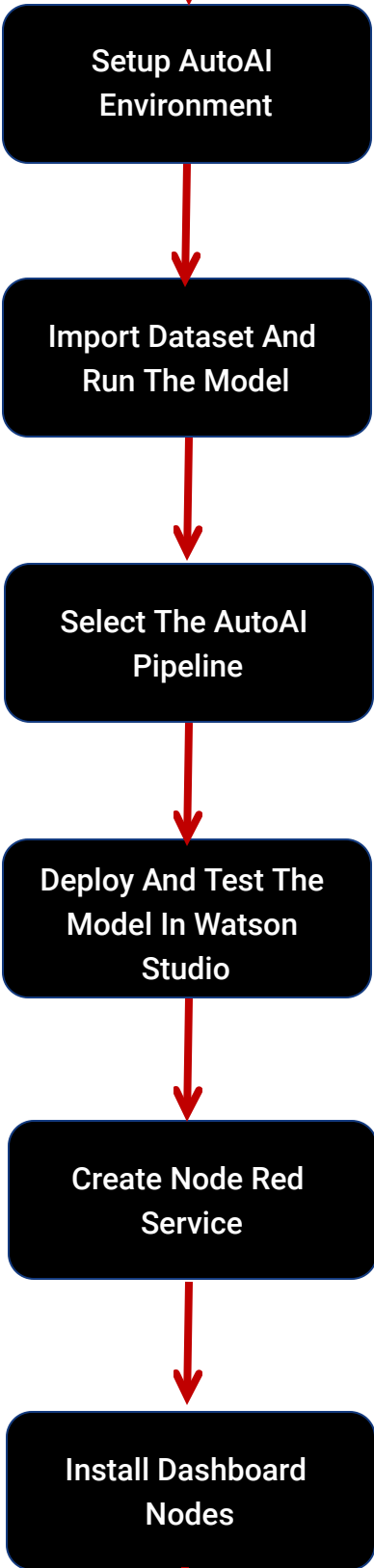
Login To IBM Cloud

Create Cloud Object
Storage

Create Watson Studio
Platform

Create Machine
Learning Service

Create Project In
Watson Studio



```
graph TD; A[Setup AutoAI Environment] --> B[Import Dataset And Run The Model]; B --> C[Select The AutoAI Pipeline]; C --> D[Deploy And Test The Model In Watson Studio]; D --> E[Create Node Red Service]; E --> F[Install Dashboard Nodes];
```

Setup AutoAI
Environment

Import Dataset And
Run The Model

Select The AutoAI
Pipeline

Deploy And Test The
Model In Watson
Studio

Create Node Red
Service

Install Dashboard
Nodes



Build UI With Nodered

Result :

The model is successfully trained and deployed using AutoAI experiment and Node Red Service. Therefore, "**The Compressive Strength Of The Concrete**", Machine Learning model can predict the compressive strength of concrete.

Advantages and Disadvantages :

➤ Advantages -

- Fast model selection. Select top-performing models in only minutes.
- Predicting the compressive strength of concrete early allows constructors to quickly understand the concrete's probable weaknesses and make a decision to manage a destruction process or continue with construction.

➤ Disadvantages -

- Sometime slow due to network glitch.
- Data must be good going in.
- Model can't be edited yet in a more granular way.

Applications :

- Using AutoAI, we can build and deploy a machine learning model with sophisticated training features and no coding. The tool does most of the work for us.
- Benefits all stakeholders as they need not wait till 28 days to know the compressive strength of the concrete.

Conclusion :

Therefore, the "**Predicting Compressive Strength of Concrete using IBM Watson AutoAI Experiment**", **Machine Learning** model is created and the purpose of the project is fulfilled.

Future Scope :

This model, right now is effective mainly for predicting the compressive strength of concrete so that constructors can easily understand the concrete probable weakness. By upgrading the dataset, this application can also be used to make predictions in other sectors.

Bibliography :

- github.com
- [smartinternz.com](https://www.smartinternz.com)
- [thesmartbridge.com](https://www.thesmartbridge.com)
- cloud.ibm.com

Appendix :

(Source code)

✓ To check for any null values in the dataset -

```
from numpy import *  
from pandas import *  
from matplotlib.pyplot import *  
import seaborn as sns  
d=read_csv('Concrete Data.csv')  
d  
d.corr()  
d.info()  
sns.heatmap(d.corr(),annot=True)  
d.isnull().any()
```

Screenshots

- Data Collection :

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Cement (cc)	Blast Furn	Fly Ash (cc)	Water (cc)	Superplas	Coarse Ag	Fine Aggr	Age (day)	Concrete compressive strength(MPa, megapascals)				
2	540	0	0	162	2.5	1040	676	28	79.99				
3	540	0	0	162	2.5	1055	676	28	61.89				
4	332.5	142.5	0	228	0	932	594	270	40.27				
5	332.5	142.5	0	228	0	932	594	365	41.05				
6	198.6	132.4	0	192	0	978.4	825.5	360	44.3				
7	266	114	0	228	0	932	670	90	47.03				
8	380	95	0	228	0	932	594	365	43.7				
9	380	95	0	228	0	932	594	28	36.45				
10	266	114	0	228	0	932	670	28	45.85				
11	475	0	0	228	0	932	594	28	39.29				
12	198.6	132.4	0	192	0	978.4	825.5	90	38.07				
13	198.6	132.4	0	192	0	978.4	825.5	28	28.02				
14	427.5	47.5	0	228	0	932	594	270	43.01				
15	190	190	0	228	0	932	670	90	42.33				
16	304	76	0	228	0	932	670	28	47.81				
17	380	0	0	228	0	932	670	90	52.91				
18	139.6	209.4	0	192	0	1047	806.9	90	39.36				
19	342	38	0	228	0	932	670	365	56.14				
20	380	95	0	228	0	932	594	90	40.56				
21	475	0	0	228	0	932	594	180	42.62				
22	427.5	47.5	0	228	0	932	594	180	41.84				
23	139.6	209.4	0	192	0	1047	806.9	28	28.24				

- Importing required libraries :

```
In [1]: from numpy import *
        from pandas import *
        from matplotlib.pyplot import *
        import seaborn as sns
```


- Importing dataset :

```
In [2]: d=read_csv('Concrete Data.csv')
d
```

Out[2]:

	Cement (component 1) (kg in a m^3 mixture)	Blast Furnace Slag (component 2)(kg in a m^3 mixture)	Fly Ash (component 3) (kg in a m^3 mixture)	Water (component 4) (kg in a m^3 mixture)	Superplasticizer (component 5)(kg in a m^3 mixture)	Coarse Aggregate (component 6)(kg in a m^3 mixture)	Fine Aggregate (component 7)(kg in a m^3 mixture)	Age (day)	Concrete compressive strength(MPa, megapascals)
0	540.0	0.0	0.0	162.0	2.5	1040.0	676.0	28	79.99
1	540.0	0.0	0.0	162.0	2.5	1055.0	676.0	28	61.89
2	332.5	142.5	0.0	228.0	0.0	932.0	594.0	270	40.27
3	332.5	142.5	0.0	228.0	0.0	932.0	594.0	365	41.05
4	198.6	132.4	0.0	192.0	0.0	978.4	825.5	360	44.30
...
1025	276.4	116.0	90.3	179.6	8.9	870.1	768.3	28	44.28
1026	322.2	0.0	115.6	196.0	10.4	817.9	813.4	28	31.18
1027	148.5	139.4	108.6	192.7	6.1	892.4	780.0	28	23.70
1028	159.1	186.7	0.0	175.6	11.3	989.6	788.9	28	32.77
1029	260.9	100.5	78.3	200.6	8.6	864.5	761.5	28	32.40

1030 rows x 9 columns

- Finding correlation between the attributes :

```
In [3]: d.corr()
```

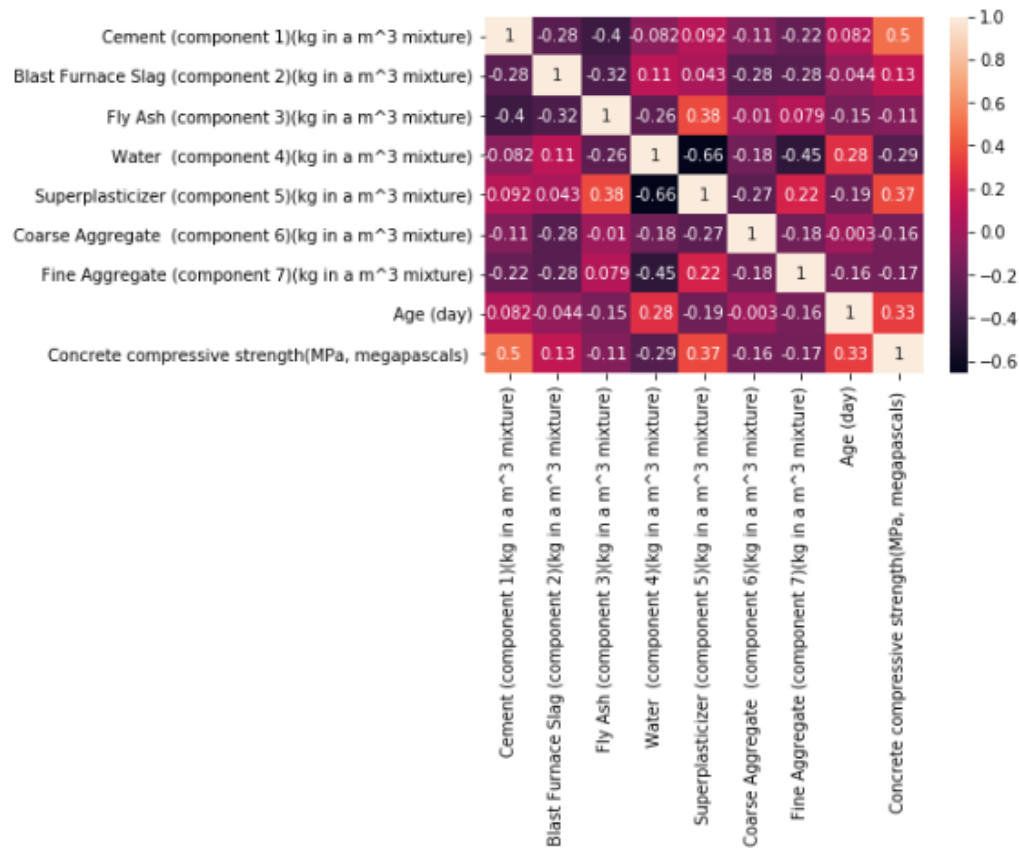
Out[3]:

	Cement (component 1)(kg in a m^3 mixture)	Blast Furnace Slag (component 2)(kg in a m^3 mixture)	Fly Ash (component 3)(kg in a m^3 mixture)	Water (component 4)(kg in a m^3 mixture)	Superplasticizer (component 5) (kg in a m^3 mixture)	Coarse Aggregate (component 6) (kg in a m^3 mixture)	Fine Aggregate (component 7) (kg in a m^3 mixture)	Age (day)	Concrete compressive strength(MPa, megapascals)
Cement (component 1)(kg in a m^3 mixture)	1.000000	-0.275216	-0.397467	-0.081587	0.092386	-0.109349	-0.222718	0.081946	0.497832
Blast Furnace Slag (component 2)(kg in a m^3 mixture)	-0.275216	1.000000	-0.323580	0.107252	0.043270	-0.283999	-0.281803	-0.044246	0.134829
Fly Ash (component 3)(kg in a m^3 mixture)	-0.397467	-0.323580	1.000000	-0.258984	0.377503	-0.009961	0.079108	-0.154371	-0.105755
Water (component 4)(kg in a m^3 mixture)	-0.081587	0.107252	-0.258984	1.000000	-0.657533	-0.182294	-0.450661	0.277618	-0.289633
Superplasticizer (component 5)(kg in a m^3 mixture)	0.092386	0.043270	0.377503	-0.657533	1.000000	-0.265999	0.222691	-0.192700	0.368079
Coarse Aggregate (component 6)(kg in a m^3 mixture)	-0.109349	-0.283999	-0.009961	-0.182294	-0.265999	1.000000	-0.178481	-0.003016	-0.164935
Fine Aggregate (component 7)(kg in a m^3 mixture)	-0.222718	-0.281803	0.079108	-0.450661	0.222691	-0.178481	1.000000	-0.156095	-0.167241
Age (day)	0.081946	-0.044246	-0.154371	0.277618	-0.192700	-0.003016	-0.156095	1.000000	0.328873
Concrete compressive strength(MPa, megapascals)	0.497832	0.134829	-0.105755	-0.289633	0.368079	-0.164935	-0.167241	0.328873	1.000000

- Data Visualisation :

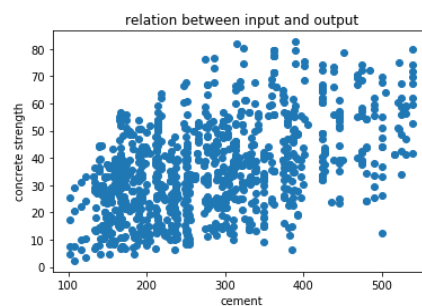
```
In [5]: sns.heatmap(d.corr(),annot=True)
```

```
Out[5]: <matplotlib.axes._subplots.AxesSubplot at 0x1ed05ebf188>
```



```
In [18]: scatter(d.iloc[:,0],y)
          xlabel('cement')
          ylabel('concrete strength')
          title('relation between input and output')

Out[18]: Text(0.5, 1.0, 'relation between input and output')
```



- Checking for null values :

```
In [6]: d.isnull().any()
```

```
Out[6]: Cement (component 1)(kg in a m^3 mixture)      False
Blast Furnace Slag (component 2)(kg in a m^3 mixture)  False
Fly Ash (component 3)(kg in a m^3 mixture)             False
Water (component 4)(kg in a m^3 mixture)               False
Superplasticizer (component 5)(kg in a m^3 mixture)    False
Coarse Aggregate (component 6)(kg in a m^3 mixture)    False
Fine Aggregate (component 7)(kg in a m^3 mixture)      False
Age (day)                                               False
Concrete compressive strength(MPa, megapascals)        False
dtype: bool
```

- Creating IBM account :

The screenshot displays the IBM Cloud Dashboard interface. At the top, there is a navigation bar with the IBM Cloud logo, a search bar, and links to Catalog, Docs, Support, and Manage. The main content area is divided into several sections. On the left, a sidebar contains icons for various services. The central part of the dashboard features a 'Resource summary' section with a list of resources and their counts, including Cloud Foundry apps, services, storage, and developer tools. To the right, there is a 'Planned maintenance' section with a graphic and a 'Recent support cases' section. The bottom of the dashboard includes a 'For you' section with personalized recommendations and a 'News' section with the latest updates from IBM.

IBM Cloud

Search resources and offerings...

Catalog Docs Support Manage Asapu Jyotsna ...

Dashboard

Upgrade Customize Create resource +

Resource summary View all

12 Resources

Cloud Foundry apps	✓ 1
Cloud Foundry services	2
Services	✓ 4
Storage	✓ 1
Apps	2
Developer tools	2

Add resources +

Planned maintenance View all

Clear skies! You can view your scheduled maintenance events here.

Recent support cases View all

For you ⓘ

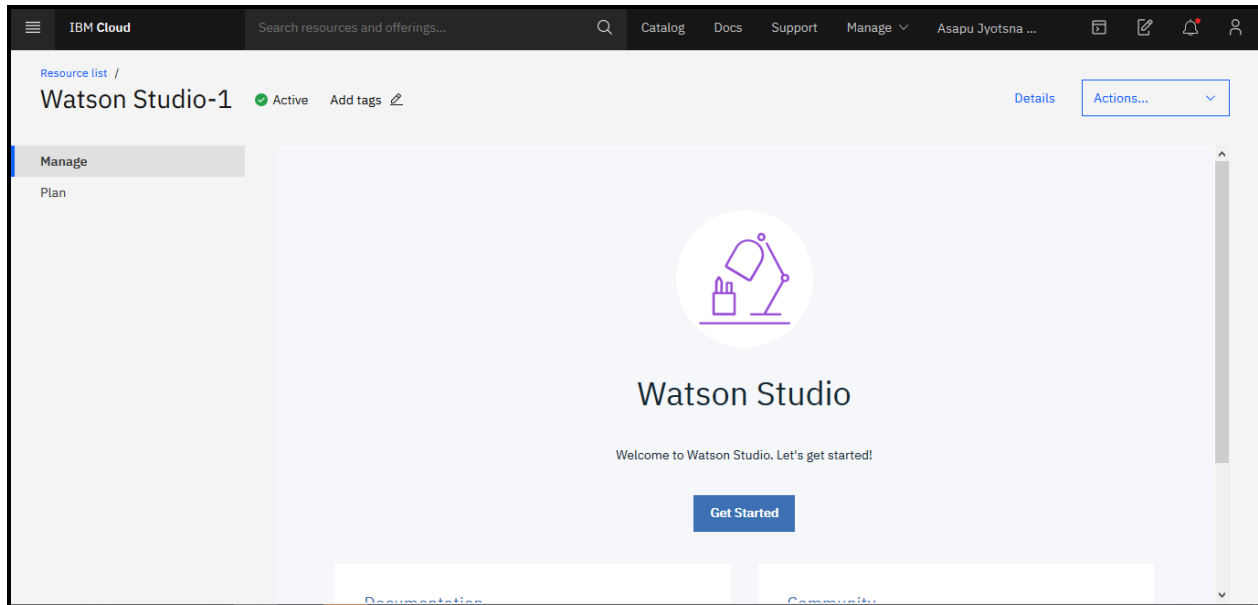
Accelerate your cloud use with starter kits. View the most popular starter kits based on use case or language

News View all

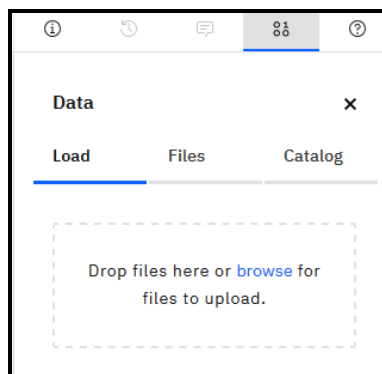
IBM CEO Arvind Krishna Keynotes IBM Think Gov Digital 2020

IBM Introduces Sterling Inventory Control Tower to Help

● Watson Studio Platform :



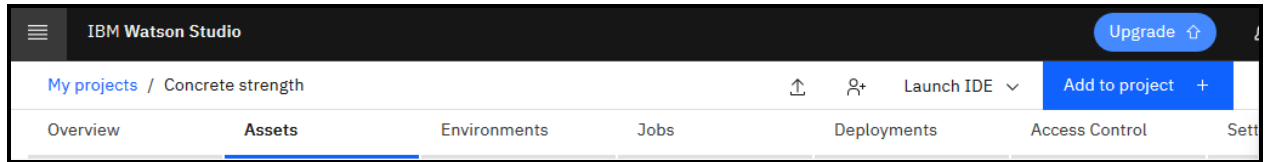
● Uploading dataset :



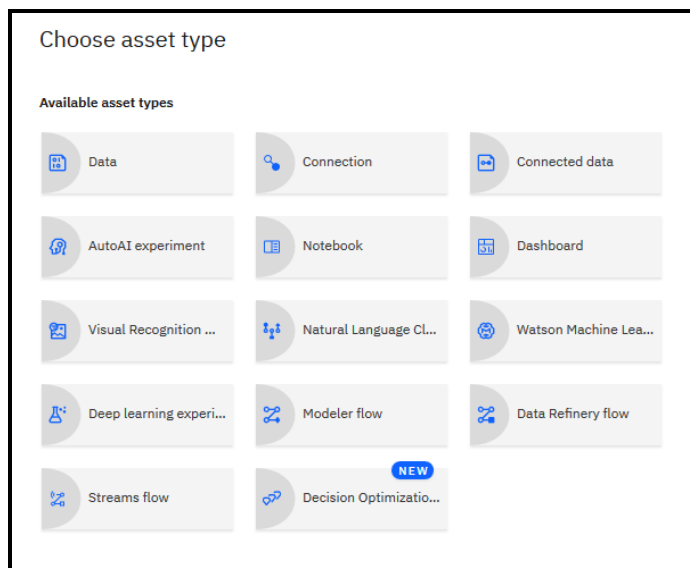
Data assets				
0 assets selected.				
<input type="checkbox"/>	Name	Type	Created by	Last modified
<input type="checkbox"/>	CSV Concrete Data.csv	Data Asset	Asapu Jyotsna Rao	Jun 30, 2020, 4:40 PM

● Creating AutoAI Environment :

- ✓ Click on Add to project -



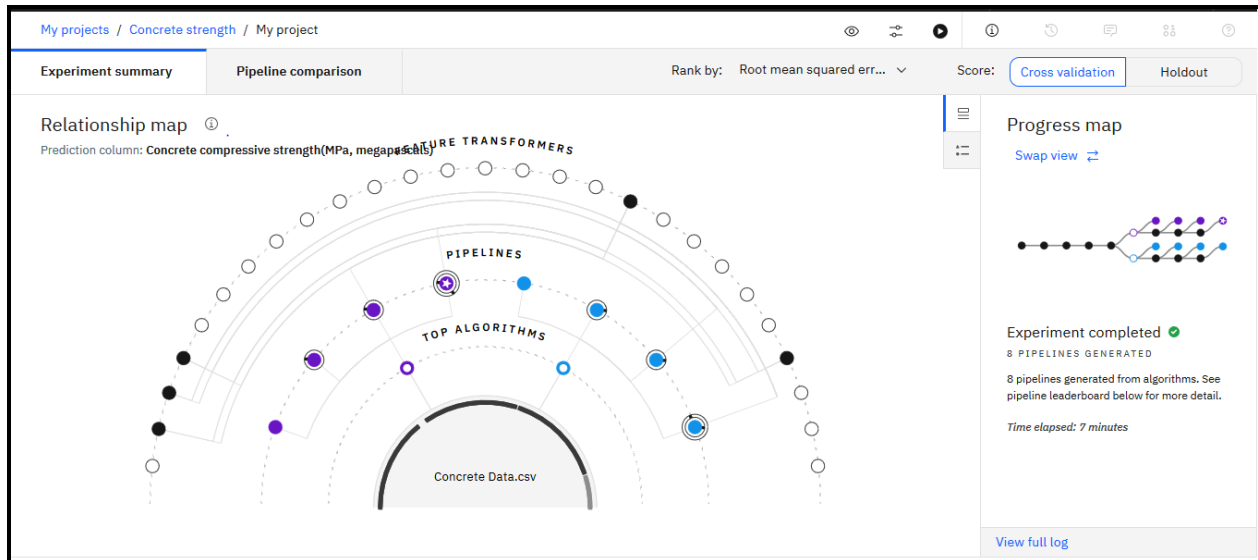
- ✓ Select AutoAI experiment as asset type -



- ✓ Create the Environment-

The screenshot shows the 'New AutoAI experiment' form. It is divided into two main sections: 'Define details' and 'Associate services'.
In the 'Define details' section, there are two tabs: 'From blank' (selected) and 'From sample'. Below the tabs, there is a 'Name *' field with the text 'project' entered, and a 'Description' field with the placeholder text 'Description of AutoAI experiment'.
In the 'Associate services' section, there is a dropdown menu for 'Watson Machine Learning Service Instance *' with 'Machine Learning-1' selected. Below this is a 'Compute configuration *' field with '8 vCPU and 32 GB RAM' selected. A note below the configuration states: 'This compute configuration consumes 20 capacity units per hour. [Learn more](#) about capacity unit hours and Watson Machine Learning pricing plans.' At the bottom right of the form, there are two buttons: 'Cancel' and 'Create'.

● Running the model :



● Pipeline Selection :

Pipeline leaderboard

	Rank	↑	Name	Algorithm	RMSE (Optimized)	Enhancements	Build time
>	★ 1		Pipeline 4	<div><div></div>Gradient Boosting Regressor</div>	4.502	<div>HPO-1FEHPO-2</div>	00:00:23
>	2		Pipeline 3	<div><div></div>Gradient Boosting Regressor</div>	4.598	<div>HPO-1FE</div>	00:02:20
>	3		Pipeline 2	<div><div></div>Gradient Boosting Regressor</div>	4.878	<div>HPO-1</div>	00:00:13
>	4		Pipeline 1	<div><div></div>Gradient Boosting Regressor</div>	5.330	<div>None</div>	00:00:01
>	5		Pipeline 7	<div><div></div>Random Forest Regressor</div>	5.471	<div>HPO-1FE</div>	00:00:45
>	6		Pipeline 8	<div><div></div>Random Forest Regressor</div>	5.471	<div>HPO-1FEHPO-2</div>	00:00:40
>	7		Pipeline 5	<div><div></div>Random Forest Regressor</div>	5.843	<div>None</div>	00:00:01

- **Deploying the model :**

- ✓ click on add deployment and deploy the model -

Model

My project - P4 GradientBoostingRegressorEstimator

Overview Evaluation **Deployments** Lineage

[Add Deployment](#) +

NAME	STATUS	TYPE	ACTIONS
concreteselection	Ready	Web Service	:

- **Testing the data :**

[My projects](#) / [Concrete strength](#) / [My project - P4 GradientBoosting...](#) / [concreteselection](#)

Overview Implementation **Test**

Enter input data

Cement (component 1)(kg in a m³ mixture)

Blast Furnace Slag (component 2)(kg in a m³ mixture)

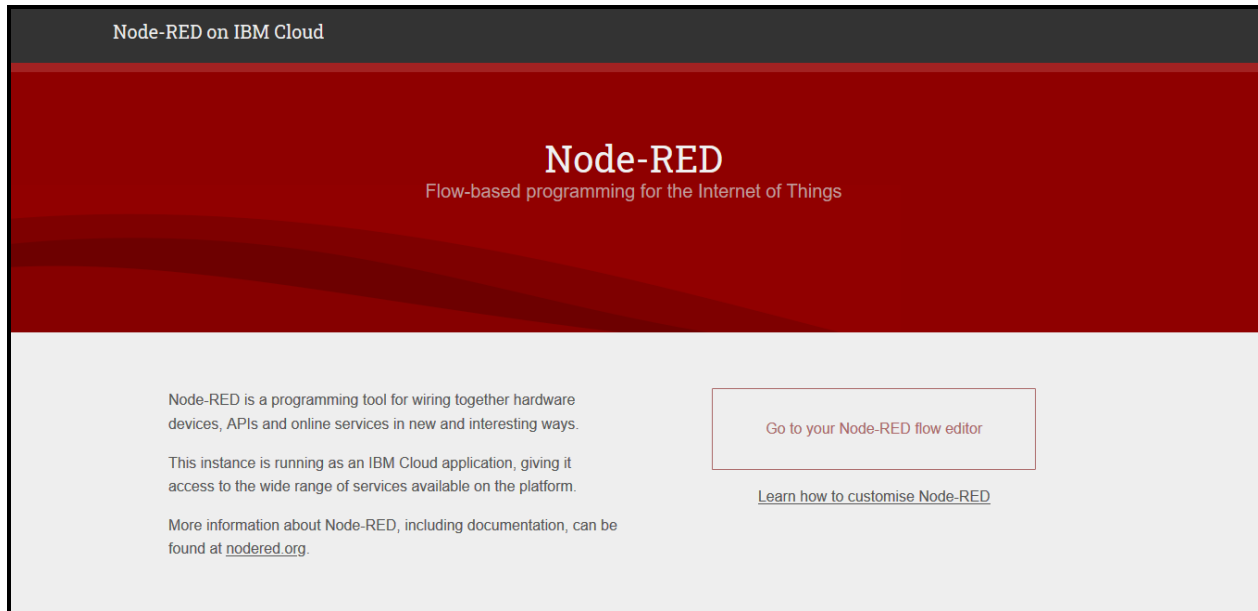
Fly Ash (component 3)(kg in a m³ mixture)

Water (component 4)(kg in a m³ mixture)

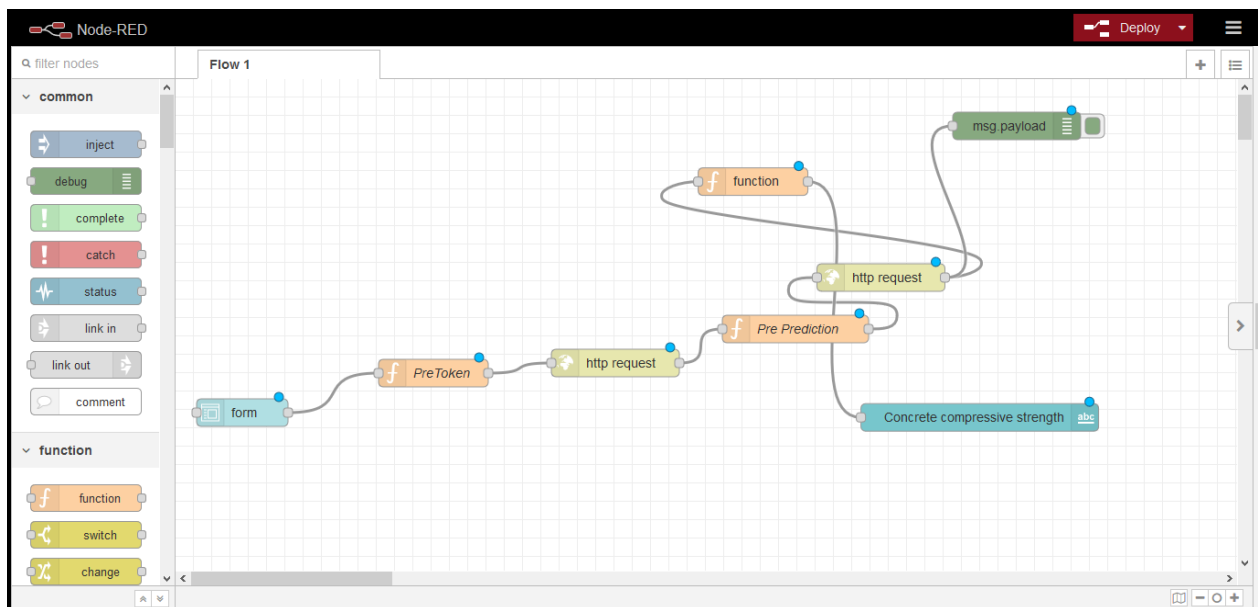
[Predict](#)

```
{
  "predictions": [
    {
      "fields": [
        "prediction"
      ],
      "values": [
        58.40117489945952
      ]
    }
  ]
}
```

- **Creating Node Red Service :**



- **Building UI with Nodered :**



● Output :

Home

Predicting Compressive Strength Of Concrete

Cement (component 1)(kg in a m³ mixture) *

540

Blast Furnace Slag (component 2)(kg in a m³ mixture) *

0

Fly Ash (component 3)(kg in a m³ mixture) *

0

Water (component 4)(kg in a m³ mixture) *

162

Superplasticizer (component 5)(kg in a m³ mixture) *

0

Coarse Aggregate (component 6)(kg in a m³ mixture) *

1040

Fine Aggregate (component 7)(kg in a m³ mixture) *

680

Age (day) *

50

SUBMIT

CANCEL

Concrete compressive strength

69.56452387399932