In [1]:

```
! pip install sagemaker botocore boto3 awscli matplotlib seaborn pandas --upgrade
```

```
Requirement already satisfied: sagemaker in /home/ec2-user/anaconda3/envs/python3/
lib/python3.10/site-packages (2.233.0)
Collecting sagemaker
  Using cached sagemaker-2.237.0-py3-none-any.whl.metadata (16 kB)
Requirement already satisfied: botocore in /home/ec2-user/anaconda3/envs/python3/l
ib/python3.10/site-packages (1.35.63)
Collecting botocore
  Using cached botocore-1.35.76-py3-none-any.whl.metadata (5.7 kB)
Requirement already satisfied: boto3 in /home/ec2-user/anaconda3/envs/python3/lib/
python3.10/site-packages (1.35.63)
Collecting boto3
  Using cached boto3-1.35.76-py3-none-any.whl.metadata (6.7 kB)
Requirement already satisfied: awscli in /home/ec2-user/anaconda3/envs/python3/li
b/python3.10/site-packages (1.36.4)
Collecting awscli
  Using cached awscli-1.36.17-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: matplotlib in /home/ec2-user/anaconda3/envs/python
3/lib/python3.10/site-packages (3.9.2)
Collecting matplotlib
  Using cached matplotlib-3.9.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x8
6_64.whl.metadata (11 kB)
Requirement already satisfied: seaborn in /home/ec2-user/anaconda3/envs/python3/li
b/python3.10/site-packages (0.13.2)
Requirement already satisfied: pandas in /home/ec2-user/anaconda3/envs/python3/li
b/python3.10/site-packages (1.5.3)
Collecting pandas
  Using cached pandas-2.2.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_6
4.whl.metadata (89 kB)
Requirement already satisfied: attrs<24,>=23.1.0 in /home/ec2-user/anaconda3/envs/
python3/lib/python3.10/site-packages (from sagemaker) (23.2.0)
Requirement already satisfied: cloudpickle==2.2.1 in /home/ec2-user/anaconda3/env
s/python3/lib/python3.10/site-packages (from sagemaker) (2.2.1)
Requirement already satisfied: docker in /home/ec2-user/anaconda3/envs/python3/li
b/python3.10/site-packages (from sagemaker) (7.1.0)
Collecting fastapi (from sagemaker)
  Using cached fastapi-0.115.6-py3-none-any.whl.metadata (27 kB)
Requirement already satisfied: google-pasta in /home/ec2-user/anaconda3/envs/pytho
n3/lib/python3.10/site-packages (from sagemaker) (0.2.0)
Requirement already satisfied: importlib-metadata<7.0,>=1.4.0 in /home/ec2-user/an
aconda3/envs/python3/lib/python3.10/site-packages (from sagemaker) (6.11.0)
Requirement already satisfied: jsonschema in /home/ec2-user/anaconda3/envs/python
3/lib/python3.10/site-packages (from sagemaker) (4.23.0)
Requirement already satisfied: numpy<2.0,>=1.9.0 in /home/ec2-user/anaconda3/envs/
python3/lib/python3.10/site-packages (from sagemaker) (1.26.4)
Collecting omegaconf<2.3,>=2.2 (from sagemaker)
  Using cached omegaconf-2.2.3-py3-none-any.whl.metadata (3.9 kB)
Requirement already satisfied: packaging>=20.0 in /home/ec2-user/anaconda3/envs/py
thon3/lib/python3.10/site-packages (from sagemaker) (21.3)
Requirement already satisfied: pathos in /home/ec2-user/anaconda3/envs/python3/li
b/python3.10/site-packages (from sagemaker) (0.3.3)
Requirement already satisfied: platformdirs in /home/ec2-user/anaconda3/envs/pytho
n3/lib/python3.10/site-packages (from sagemaker) (4.3.6)
Requirement already satisfied: protobuf<5.0,>=3.12 in /home/ec2-user/anaconda3/env
s/python3/lib/python3.10/site-packages (from sagemaker) (4.25.5)
Requirement already satisfied: psutil in /home/ec2-user/anaconda3/envs/python3/li
b/python3.10/site-packages (from sagemaker) (6.0.0)
```

```
Requirement already satisfied: pyyaml~=6.0 in /home/ec2-user/anaconda3/envs/python
3/lib/python3.10/site-packages (from sagemaker) (6.0.2)
Requirement already satisfied: requests in /home/ec2-user/anaconda3/envs/python3/l
ib/python3.10/site-packages (from sagemaker) (2.32.3)
Collecting sagemaker-core<2.0.0,>=1.0.17 (from sagemaker)
  Using cached sagemaker_core-1.0.17-py3-none-any.whl.metadata (4.9 kB)
Requirement already satisfied: schema in /home/ec2-user/anaconda3/envs/python3/li
b/python3.10/site-packages (from sagemaker) (0.7.7)
Requirement already satisfied: smdebug-rulesconfig==1.0.1 in /home/ec2-user/anacon
da3/envs/python3/lib/python3.10/site-packages (from sagemaker) (1.0.1)
Requirement already satisfied: tblib<4,>=1.7.0 in /home/ec2-user/anaconda3/envs/py
thon3/lib/python3.10/site-packages (from sagemaker) (3.0.0)
Requirement already satisfied: tqdm in /home/ec2-user/anaconda3/envs/python3/lib/p
ython3.10/site-packages (from sagemaker) (4.66.5)
Requirement already satisfied: urllib3<3.0.0,>=1.26.8 in /home/ec2-user/anaconda3/
envs/python3/lib/python3.10/site-packages (from sagemaker) (2.2.3)
Collecting uvicorn (from sagemaker)
  Using cached uvicorn-0.32.1-py3-none-any.whl.metadata (6.6 kB)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /home/ec2-user/anaconda3/
envs/python3/lib/python3.10/site-packages (from botocore) (1.0.1)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /home/ec2-user/anaco
nda3/envs/python3/lib/python3.10/site-packages (from botocore) (2.9.0)
Requirement already satisfied: s3transfer<0.11.0,>=0.10.0 in /home/ec2-user/anacon
da3/envs/python3/lib/python3.10/site-packages (from boto3) (0.10.3)
Requirement already satisfied: docutils<0.17,>=0.10 in /home/ec2-user/anaconda3/en
vs/python3/lib/python3.10/site-packages (from awscli) (0.16)
Requirement already satisfied: colorama<0.4.7,>=0.2.5 in /home/ec2-user/anaconda3/
envs/python3/lib/python3.10/site-packages (from awscli) (0.4.6)
Requirement already satisfied: rsa<4.8,>=3.1.2 in /home/ec2-user/anaconda3/envs/py
thon3/lib/python3.10/site-packages (from awscli) (4.7.2)
Requirement already satisfied: contourpy>=1.0.1 in /home/ec2-user/anaconda3/envs/p
ython3/lib/python3.10/site-packages (from matplotlib) (1.3.0)
Requirement already satisfied: cycler>=0.10 in /home/ec2-user/anaconda3/envs/pytho
n3/lib/python3.10/site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /home/ec2-user/anaconda3/envs/
python3/lib/python3.10/site-packages (from matplotlib) (4.54.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /home/ec2-user/anaconda3/envs/
python3/lib/python3.10/site-packages (from matplotlib) (1.4.7)
Requirement already satisfied: pillow>=8 in /home/ec2-user/anaconda3/envs/python3/
lib/python3.10/site-packages (from matplotlib) (11.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in /home/ec2-user/anaconda3/envs/p
ython3/lib/python3.10/site-packages (from matplotlib) (3.2.0)
Requirement already satisfied: pytz>=2020.1 in /home/ec2-user/anaconda3/envs/pytho
n3/lib/python3.10/site-packages (from pandas) (2024.2)
Collecting tzdata>=2022.7 (from pandas)
  Using cached tzdata-2024.2-py2.py3-none-any.whl.metadata (1.4 kB)
Requirement already satisfied: zipp>=0.5 in /home/ec2-user/anaconda3/envs/python3/
lib/python3.10/site-packages (from importlib-metadata<7.0,>=1.4.0->sagemaker) (3.2
0.2)
Collecting antlr4-python3-runtime==4.9.* (from omegaconf<2.3,>=2.2->sagemaker)
  Using cached antlr4-python3-runtime-4.9.3.tar.gz (117 kB)
  Preparing metadata (setup.py) ... error
  error: subprocess-exited-with-error

  × python setup.py egg_info did not run successfully.
  │ exit code: 1
  ╰─> [43 lines of output]
      running egg_info
      creating /tmp/pip-pip-egg-info-ucldx2_t/antlr4_python3_runtime.egg-info
      writing /tmp/pip-pip-egg-info-ucldx2_t/antlr4_python3_runtime.egg-info/PKG-I
NFO
      writing dependency_links to /tmp/pip-pip-egg-info-ucldx2_t/antlr4_python3_ru
ntime.egg-info/dependency_links.txt
      writing requirements to /tmp/pip-pip-egg-info-ucldx2_t/antlr4_python3_runtim
e.egg-info/requires.txt
      writing top-level names to /tmp/pip-pip-egg-info-ucldx2_t/antlr4_python3_run
time.egg-info/top_level.txt
      writing manifest file '/tmp/pip-pip-egg-info-ucldx2_t/antlr4_python3_runtim
e.egg-info/SOURCES.txt'
      reading manifest file '/tmp/pip-pip-egg-info-ucldx2_t/antlr4_python3_runtim
e.egg-info/SOURCES.txt'
      reading manifest template 'MANIFEST.in'
      warning: no files found matching '*.py' under directory 'test'
      warning: no files found matching '*.c' under directory 'test'
      Traceback (most recent call last):
        File "<string>", line 2, in <module>
        File "<pip-setuptools-caller>", line 34, in <module>
        File "/tmp/pip-install-c3qupc_6/antlr4-python3-runtime_99a3c34b438a48f6a45
f14779e9959e6/setup.py", line 3, in <module>
          setup(
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/__init__.py", line 117, in setup
          return distutils.core.setup(**attrs)
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/_distutils/core.py", line 183, in setup
          return run_commands(dist)
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/_distutils/core.py", line 199, in run_commands
          dist.run_commands()
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/_distutils/dist.py", line 954, in run_commands
          self.run_command(cmd)
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/dist.py", line 950, in run_command
          super().run_command(command)
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/_distutils/dist.py", line 973, in run_command
          cmd_obj.run()
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/command/egg_info.py", line 311, in run
          self.find_sources()
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/command/egg_info.py", line 319, in find_sources
          mm.run()
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/command/egg_info.py", line 545, in run
          self.prune_file_list()
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/command/sdist.py", line 161, in prune_file_list
          super().prune_file_list()
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/_distutils/command/sdist.py", line 380, in prune_file_list
```

```
          base_dir = self.distribution.get_fullname()
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/_core_metadata.py", line 267, in get_fullname
          return _distribution_fullname(self.get_name(), self.get_version())
        File "/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/s
etuptools/_core_metadata.py", line 285, in _distribution_fullname
          canonicalize_version(version, strip_trailing_zero=False),
      TypeError: canonicalize_version() got an unexpected keyword argument 'strip_
trailing_zero'
      [end of output]

  note: This error originates from a subprocess, and is likely not a problem with
pip.
error: metadata-generation-failed

× Encountered error while generating package metadata.
╰─> See above for output.

note: This is an issue with the package mentioned above, not pip.
hint: See above for details.
```

In [2]:

```python
import sagemaker
import json
import pandas as pd
import numpy as np
import boto3
import seaborn as sns
import matplotlib.pyplot as plt
import time
from sagemaker import get_execution_role
import warnings as warnings
warnings.filterwarnings("ignore")
from sklearn.model_selection import train_test_split
# Importing the MinMaxScaler
from sklearn.preprocessing import MinMaxScaler
%matplotlib inline
```

```
sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemake
r/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /home/ec2-user/.c
onfig/sagemaker/config.yaml
```

In [3]:

```python
sagemaker_session = sagemaker.Session() # create  a SageMaker session
s3 = sagemaker_session.boto_session.resource("s3") #creates a resource object for i

region = boto3.Session().region_name
role = sagemaker.get_execution_role()

!aws s3 cp s3://diabetes-hb/diabetes_dataset.csv .
```

```
download: s3://diabetes-hb/diabetes_dataset.csv to ./diabetes_dataset.csv
```

In [4]:

```python
data = pd.read_csv('diabetes_dataset.csv')
data.head()
```

Out[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	A
0	6	148	72	35	0	33.6	0.627	
1	1	85	66	29	0	26.6	0.351	
2	8	183	64	0	0	23.3	0.672	
3	1	89	66	23	94	28.1	0.167	
4	0	137	40	35	168	43.1	2.288	

In [5]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [6]:

```
data.shape
```

Out[6]:

```
(768, 9)
```

Count nulls in each column

In [7]:

```
# The columns that can have a 0 as an input are pregnancies and outcome
# The dataset has 0's in replace for null in the rest of the columns so I need
# to clean that and put corresponding numbers instead.
data2 = data[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'Diabe

# Count nulls (including zeros treated as null) for null_columns
null_counts = data2.isnull().sum() + (data2 == 0).sum()
print(null_counts)
```

```
Glucose                       5
BloodPressure                35
SkinThickness               227
Insulin                     374
BMI                          11
DiabetesPedigreeFunction      0
Age                           0
dtype: int64
```

Replace missing values with means

In [8]:

```
# Clean Glucose Column
mean_Glucose = data['Glucose'][data['Glucose'] != 0].mean()
data.loc[data['Glucose'] == 0, 'Glucose'] = mean_Glucose

# Clean Blood Pressure Column
mean_BloodPressure = data['BloodPressure'][data['BloodPressure'] != 0].mean()
data.loc[data['BloodPressure'] == 0, 'BloodPressure'] = mean_BloodPressure

# Clean Skin Thickness Column
mean_SkinThickness = data['SkinThickness'][data['SkinThickness'] != 0].mean()
data.loc[data['SkinThickness'] == 0, 'SkinThickness'] = mean_SkinThickness

# Clean Insulin Column
mean_Insulin = data['Insulin'][data['Insulin'] != 0].mean()
data.loc[data['Insulin'] == 0, 'Insulin'] = mean_Insulin

# Clean BMI Column
mean_BMI = data['BMI'][data['BMI'] != 0].mean()
data.loc[data['BMI'] == 0, 'BMI'] = mean_BMI

print(f'TRANSFORMATION 1: Handling missing values\nDATA ANALYSIS 1: Calculating mea
print(f'Means:\nBlood Pressure:{mean_BloodPressure}\nGlucose:{mean_Glucose}\nInsuli
print(f'\n I replaced all missing values with the means of their respective columns
```

```
TRANSFORMATION 1: Handling missing values
DATA ANALYSIS 1: Calculating means

Means:
Blood Pressure:72.40518417462484
Glucose:121.6867627785059
Insulin:155.5482233502538
BMI:32.457463672391015
Skin Thickness:29.153419593345657

 I replaced all missing values with the means of their respective columns.
```

In [9]:

```
# Need to run the null counter again to see if it was changed, had to run entire co

data2 = data[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'Diabe
null_counts = data2.isnull().sum() + (data2 == 0).sum()
print(null_counts)
```

```
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
dtype: int64
```

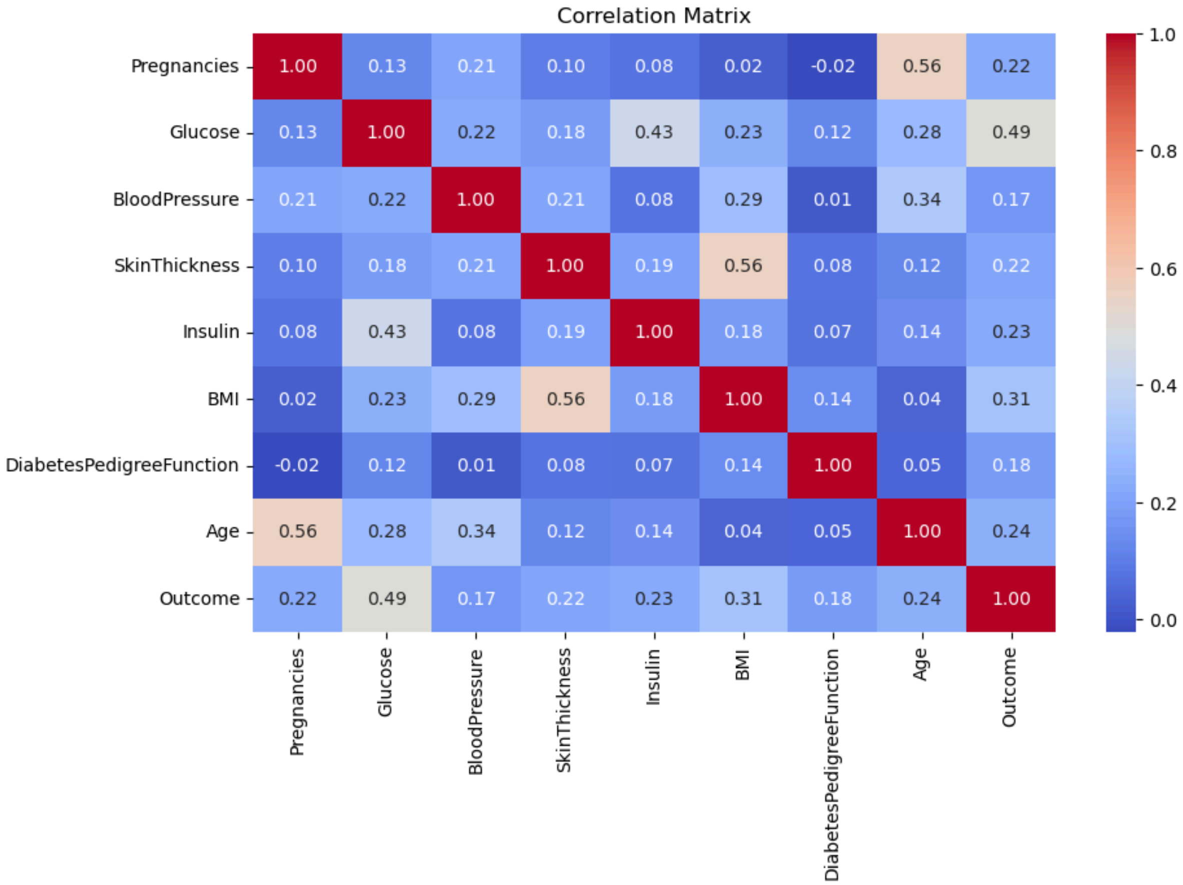
In [10]:

```
#Clip outliers
columns_to_clip = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Ins
original_data = data[columns_to_clip].copy()

clipped_columns = {}
for col in columns_to_clip:
    lower_bound = data[col].quantile(0.02)
    upper_bound = data[col].quantile(0.98)
    clipped_data = data[col].clip(lower=lower_bound, upper=upper_bound)
    data[col] = clipped_data
    clipped_columns[col] = original_data[col][(original_data[col] < lower_bound) |


print(f'TRANSFORMATION 2: Clipping any data outliers to stabilize my future models
print("Clipped values by column:")
for col, clipped in clipped_columns.items():
    if not clipped.empty:
        print(f"{col}: {clipped.dropna().values}")

print(f'\nI clipped all outliers that were in the 2nd and 98th percentile, so only
TRANSFORMATION 2: Clipping any data outliers to stabilize my future models perform
ances
TRANSFORMATION 3: Data duplication - to see if there were any outliers clipped

Clipped values by column:
Pregnancies: [13 13 13 15 17 13 14 13 13 14 13 13 13 13]
Glucose: [197. 196.  71.  44.  62.  71.  57. 194. 196. 197. 193.  71. 194.  61.
 196. 193.  72. 197.  71. 194. 195.  68.  57. 198. 197.  67.  68. 199.
  68. 195.  56.  65.]
BloodPressure: [ 40.  30. 110.  48.  44. 108.  48. 122.  48.  30. 110. 104.  48.
46.
 108. 102. 100. 100.  48. 104. 110.  44.  44.  24.  38. 106. 106. 106.
 100. 114.  46.  44.]
SkinThickness: [11. 11. 10. 60. 54. 51. 56. 50. 54.  7. 50. 52. 10. 10. 11.  8. 4
9. 11.
  8. 63. 10.  7. 52. 49. 99. 11. 50. 10. 49. 11.]
Insulin: [543. 846.  36.  23.  18.  36. 495.  37. 485.  23. 495. 478.  32. 744.
  37. 680. 545.  29. 579. 474.  14.  36. 480.  18. 600.  25.  15. 540.
 480.  22. 510.  16.]
BMI: [19.9 19.4 19.6 48.8 19.1 49.7 53.2 55.  47.9 50.  67.1 52.3 18.4 52.3
 52.9 19.3 47.9 48.3 20.  18.2 18.2 59.4 19.6 19.6 18.2 19.5 20.1 19.5
 57.3 49.6 49.3]
DiabetesPedigreeFunction: [2.288 1.441 1.893 1.781 0.102 0.088 0.096 1.4   0.085
0.084 0.101 2.329
 0.089 0.092 0.078 1.391 1.476 2.137 1.731 1.6   0.108 2.42  0.107 0.085
 1.699 0.088 0.1   1.698 1.461 0.115 1.394 0.118]
Age: [69 65 66 65 65 67 72 81 67 66 67 66 70 68 69 66]

I clipped all outliers that were in the 2nd and 98th percentile, so only the very
far outliers.
```

In [11]:

```python
#Correlation matrix heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(data.corr(), annot=True, fmt=".2f", cmap='coolwarm')
plt.title('Correlation Matrix')
print(f'DATA ANALYSIS 2: Correlation matrix heatmap')
plt.show()
print(f'\nThis showed me the correlation of all columns with aall columns to see wh
```

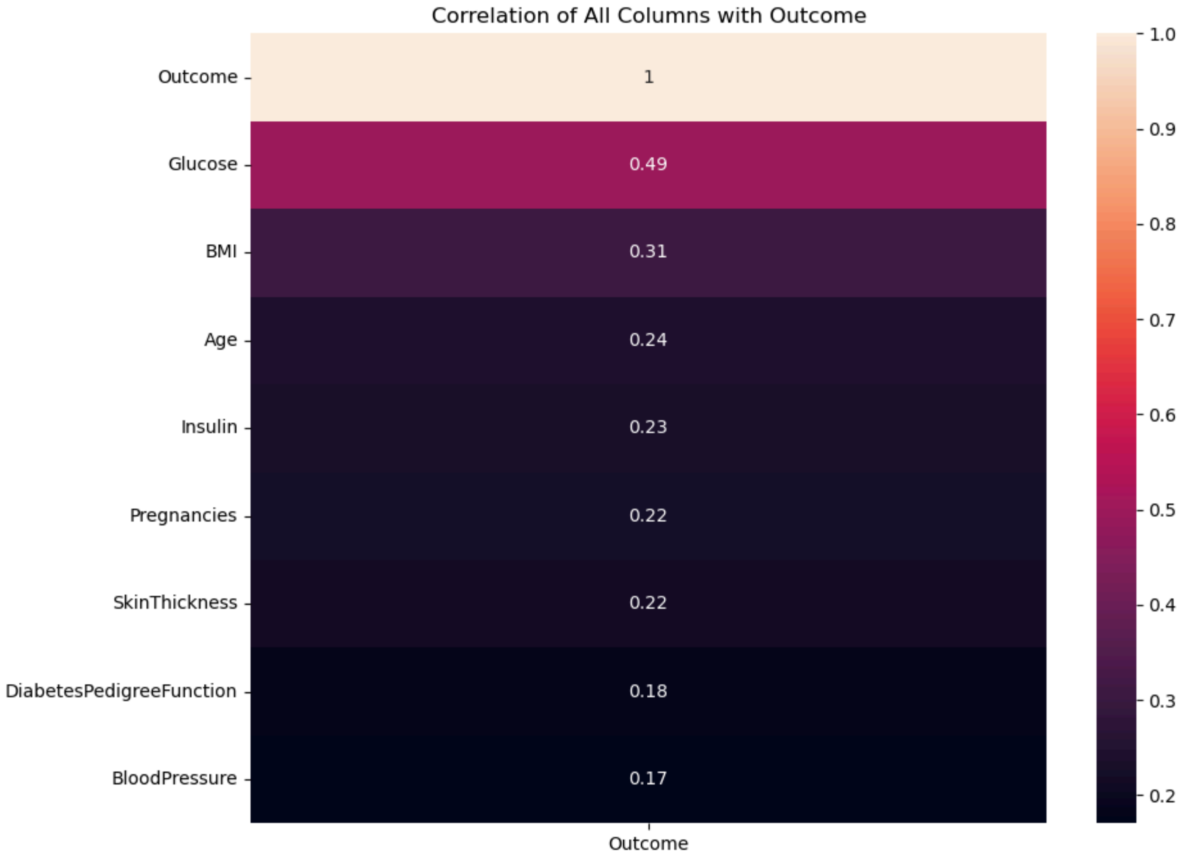
```
DATA ANALYSIS 2: Correlation matrix heatmap
```



This showed me the correlation of all columns with aall columns to see which ones had some sort of correlation, to see which ones I would like to analyze.

In [12]:
```
# Correlation with outcome
corr_with_outcome = data.corr()['Outcome'].sort_values(ascending=False)

plt.figure(figsize=(10, 8))
sns.heatmap(corr_with_outcome.to_frame(), annot=True)
plt.title('Correlation of All Columns with Outcome')
print(f'DATA ANALYSIS 3: Correlation with just outcome to see clearly')
plt.show()

print(f'This made me realize the columns with the highest correlations to diagnosis
```

```
DATA ANALYSIS 3: Correlation with just outcome to see clearly
```

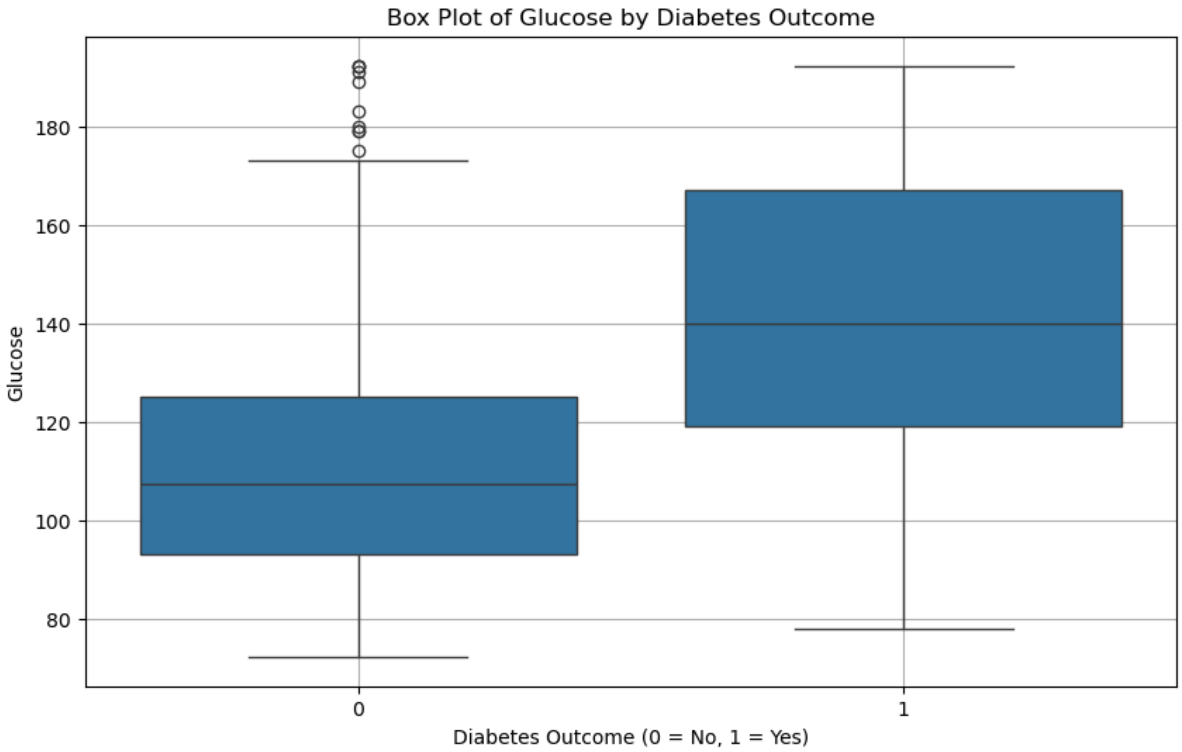


```
This made me realize the columns with the highest correlations to diagnosis are gl
ucose, BMI, and Age.
```

In [13]:

```python
#Comparing Glucose to Outcome
plt.figure(figsize=(10, 6))
sns.boxplot(data=data, x='Outcome', y='Glucose')
plt.title('Box Plot of Glucose by Diabetes Outcome')
plt.xlabel('Diabetes Outcome (0 = No, 1 = Yes)')
plt.ylabel('Glucose')
plt.grid()
print('DATA ANALYSIS 6: Visualizing the correlation between Outcome and Glucose lev
plt.show()

print('This showed me that your chances of having diabetes is higher if your have h
```

```
DATA ANALYSIS 6: Visualizing the correlation between Outcome and Glucose levels
```



```
This showed me that your chances of having diabetes is higher if your have higher
Glucose levels.
```

In [14]:

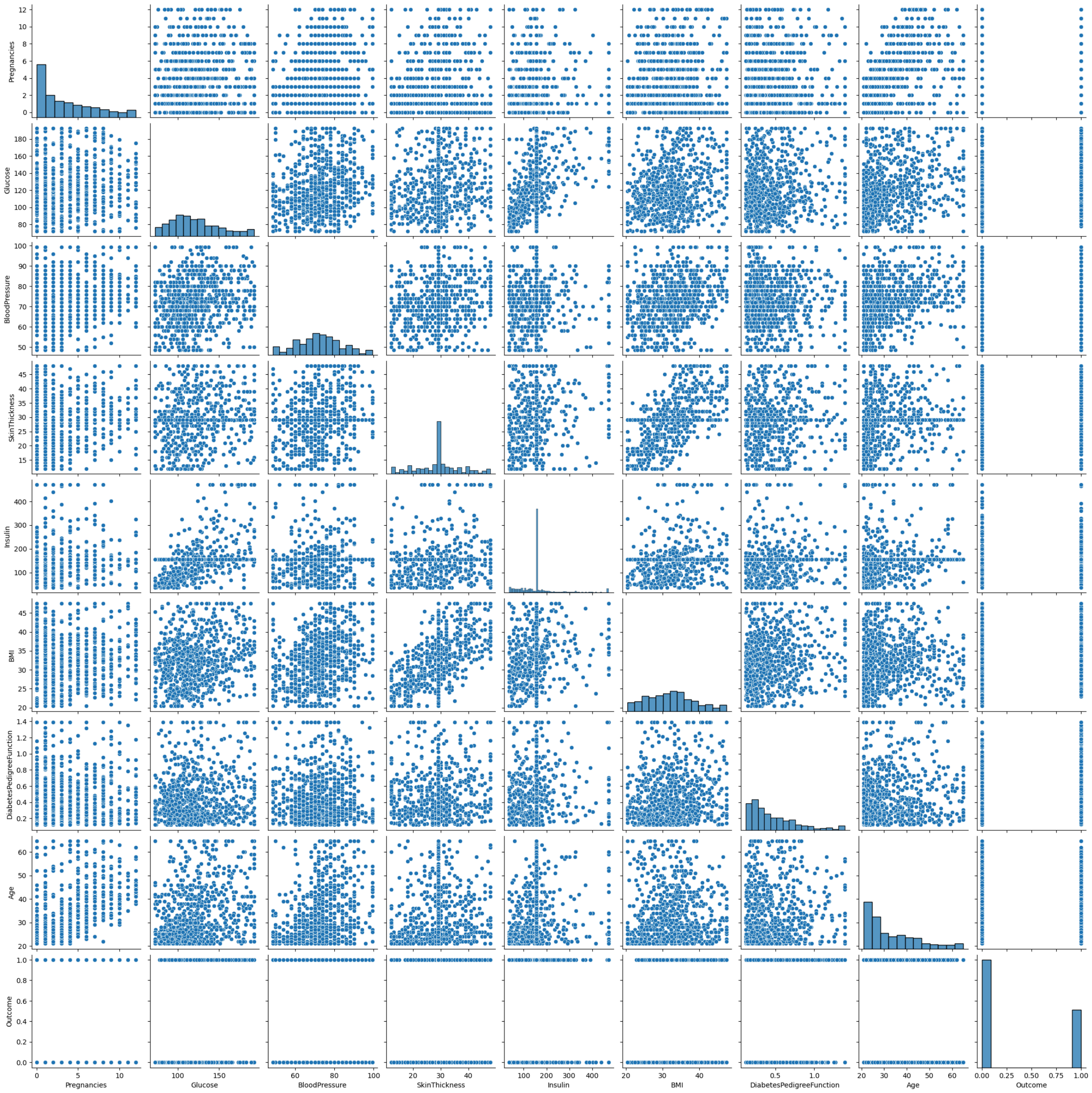
```
data.describe()
```

Out[14]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPe
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	
mean	3.816406	121.783846	72.389559	29.044045	153.909525	32.375193	
std	3.289015	29.951327	11.277319	8.104665	74.726459	6.516605	
min	0.000000	72.340000	48.680000	12.000000	37.340000	20.400000	
25%	1.000000	99.750000	64.000000	25.000000	121.500000	27.500000	
50%	3.000000	117.000000	72.202592	29.153420	155.548223	32.400000	
75%	6.000000	140.250000	80.000000	32.000000	155.548223	36.600000	
max	12.000000	192.320000	99.320000	48.000000	470.940000	47.526000	

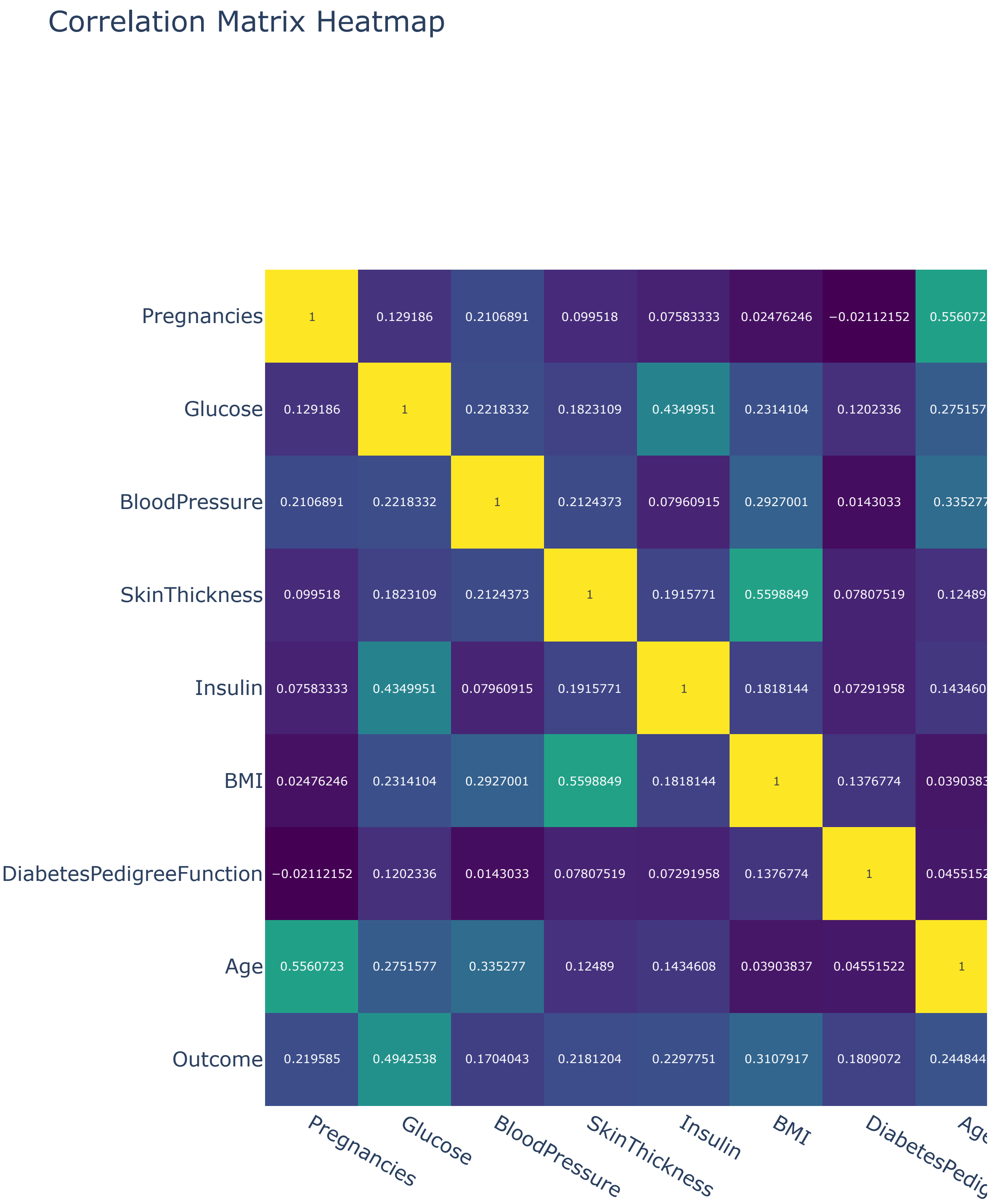
In [15]:

```
#Just to visualize which features have most correlation
# High correlation with Insuil and BMI and insulin and glucose
sns.pairplot(data)
plt.show()
```



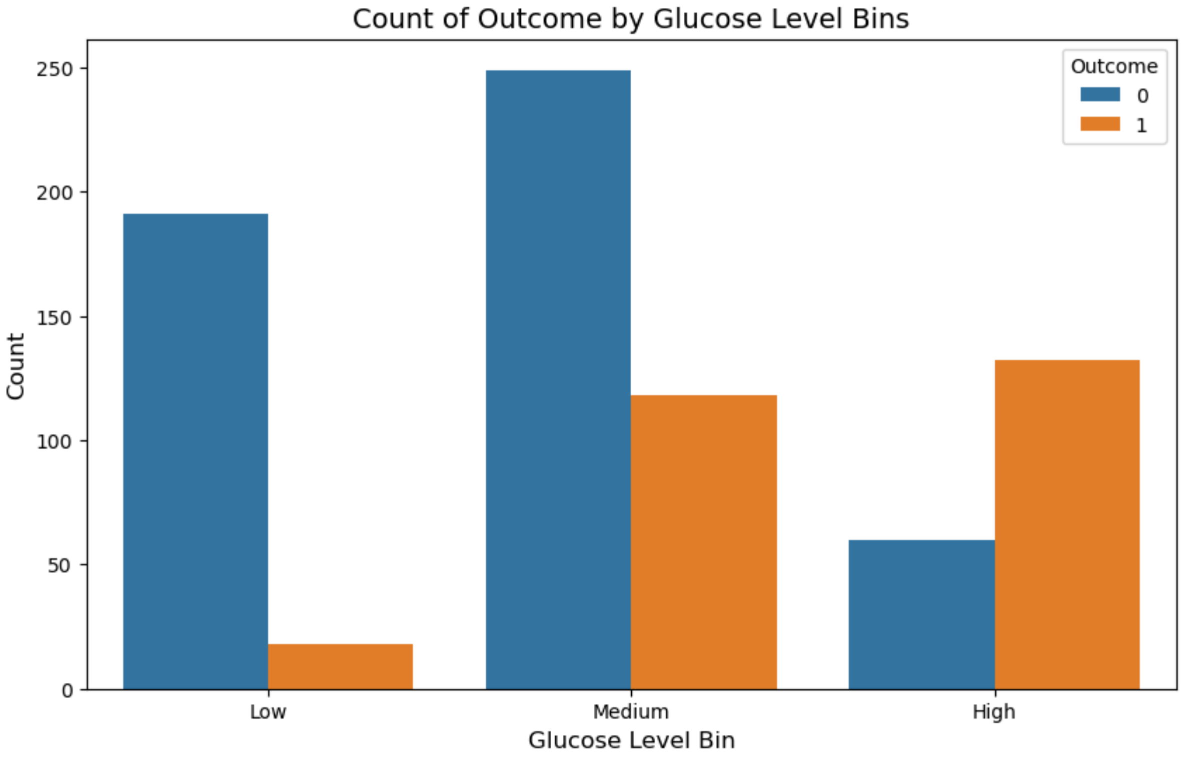
In [41]:

```
# A different visualziation for the correlation of all the columns between eachothe
import pandas as pd
import plotly.express as px

correlation_matrix = data.corr()



fig = px.imshow(
    correlation_matrix,
    text_auto=True,
    color_continuous_scale='Viridis',
    title="Correlation Matrix Heatmap"
)

fig.update_layout(
    coloraxis_colorbar=dict(
        title="Correlation",
        thickness=15,
        len=0.7,
        x=1.05,
        y=0.5,
    ),
    width=800,
    height=800,
    margin=dict(l=20, r=20, t=50, b=20),
)

fig.show()
```



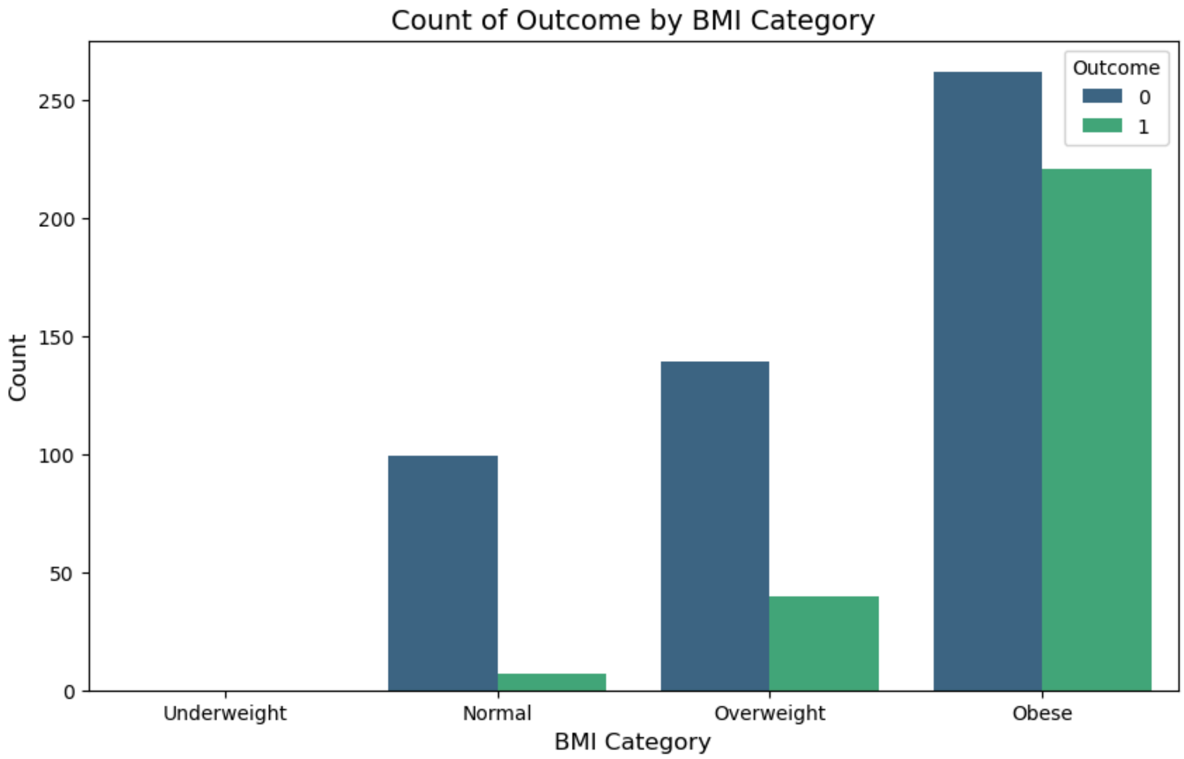
In [90]:
```
#A vizualiaztion for the highest correlation column with outcome which is glucose
plt.figure(figsize=(10, 6))

# Using binned glucose values (group into Low, Medium, High)
sns.countplot(data=data,
              x=pd.cut(data['Glucose'], bins=[0, 100, 140, float('inf')],
                       labels=['Low', 'Medium', 'High']),
              hue='Outcome')
```

```
plt.title('Count of Outcome by Glucose Level Bins', fontsize=14)
plt.xlabel('Glucose Level Bin', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.show()
```



In [93]:

```
#A vizualiaztion for outcome with BMI
plt.figure(figsize=(10, 6))

# Plot BMI categories vs Outcome
sns.countplot(data=data,
              x=pd.cut(data['BMI'], bins=[0, 18.5, 24.9, 29.9, float('inf')],
                       labels=['Underweight', 'Normal', 'Overweight', 'Obese']),
              hue='Outcome', palette='viridis')

plt.title('Count of Outcome by BMI Category', fontsize=14)
plt.xlabel('BMI Category', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.show()
```



Training

In [16]:

```
from sklearn.model_selection import train_test_split

X = data.drop(columns=['Outcome'])
y = data['Outcome']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_sta

X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.1,
```

In [17]:

```
X_train.head()
```

Out[17]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFuncti
703	2	129.0	72.405184	29.15342	155.548223	38.5	0.3
620	2	112.0	86.000000	42.00000	160.000000	38.4	0.2
337	5	115.0	76.000000	29.15342	155.548223	31.2	0.3
252	2	90.0	80.000000	14.00000	55.000000	24.4	0.2
441	2	83.0	66.000000	23.00000	50.000000	32.2	0.4

In [18]:

```
X_test.head()
```

Out[18]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFuncti
661	1	192.32	76.0	43.00000	155.548223	42.9	1.390
122	2	107.00	74.0	30.00000	100.000000	33.6	0.404
113	4	76.00	62.0	29.15342	155.548223	34.0	0.391
14	5	166.00	72.0	19.00000	175.000000	25.8	0.587
529	0	111.00	65.0	29.15342	155.548223	24.6	0.660

In [19]:

```
X_val.head(5)
```

Out[19]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFuncti
391	5	166.0	76.00	29.15342	155.548223	45.7	0.3
611	3	174.0	58.00	22.00000	194.000000	32.9	0.5
427	1	181.0	64.00	30.00000	180.000000	34.1	0.3
43	9	171.0	99.32	24.00000	240.000000	45.4	0.7
192	7	159.0	66.00	29.15342	155.548223	30.4	0.3

In [20]:

```
import pandas as pd
training_data = pd.DataFrame({**X_train, 'Outcome': y_train})
training_data.to_csv('diabetes-training_data.csv', header=False, index=False)
training_data.head()
```

Out[20]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFuncti
703	2	129.0	72.405184	29.15342	155.548223	38.5	0.3
620	2	112.0	86.000000	42.00000	160.000000	38.4	0.2
337	5	115.0	76.000000	29.15342	155.548223	31.2	0.3
252	2	90.0	80.000000	14.00000	55.000000	24.4	0.2
441	2	83.0	66.000000	23.00000	50.000000	32.2	0.4

In [21]:

```
s3_bucket = 'diabetes-hb'
prefix = 'MyModel'
!aws s3 cp diabetes-training_data.csv s3://{s3_bucket}/{prefix}/input/diabetes-trai
```

```
upload: ./diabetes-training_data.csv to s3://diabetes-hb/MyModel/input/diabetes-tr
aining_data.csv
```

In [22]:

```
validation_data = pd.DataFrame({**X_val, 'Outcome': y_val})
validation_data.to_csv('diabetes-validation_data.csv',header=False, index=False)
validation_data.head()
```

Out[22]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFuncti
391	5	166.0	76.00	29.15342	155.548223	45.7	0.3
611	3	174.0	58.00	22.00000	194.000000	32.9	0.5
427	1	181.0	64.00	30.00000	180.000000	34.1	0.3
43	9	171.0	99.32	24.00000	240.000000	45.4	0.7
192	7	159.0	66.00	29.15342	155.548223	30.4	0.3

In [23]:

```
s3_bucket = 'diabetes-hb'
prefix = 'MyModel'
!aws s3 cp diabetes-validation_data.csv s3://{s3_bucket}/{prefix}/input/diabetes-va
```

```
upload: ./diabetes-validation_data.csv to s3://diabetes-hb/MyModel/input/diabetes-
validation_data.csv
```

In [24]:

```
testing_data = pd.DataFrame({**X_test, 'Outcome': y_test})
testing_data = testing_data.to_csv('diabetes-testing_data.csv', header=False,index=
s3_bucket = 'diabetes-hb'
prefix = 'MyModel'
!aws s3 cp diabetes-testing_data.csv s3://{s3_bucket}/{prefix}/input/diabetes-testi
```

```
upload: ./diabetes-testing_data.csv to s3://diabetes-hb/MyModel/input/diabetes-tes
ting_data.csv
```

In [25]:

```
#created a sagemaker session - create a resource object for interacting with S3
from sagemaker import get_execution_role

role = get_execution_role()
session = sagemaker.Session()
region_name = boto3.Session().region_name
smclient = boto3.Session().client("sagemaker")
```

In [26]:

```
# created a variable to show the location for input and output of s3 bucket for tra
training_s3_input_location = f"s3://{s3_bucket}/{prefix}/input/diabetes-training_da
training_s3_output_location = f"s3://{s3_bucket}/{prefix}/output/"
```

In [27]:

```
validation_s3_input_location = f"s3://{s3_bucket}/{prefix}/input/diabetes-validatio
validation_s3_output_location = f"s3://{s3_bucket}/{prefix}/output/"
```

In [28]:

```
#Training channel
from sagemaker.inputs import TrainingInput
train = TrainingInput(training_s3_input_location, content_type="text/csv")
```

In [29]:

```
#get container image
from sagemaker.image_uris import retrieve
container = retrieve(framework="xgboost", region=region_name, version="1.5-1")
container
```

Out[29]:

```
'683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboost:1.5-1'
```

In [30]:

```
from time import gmtime, strftime, sleep

tuning_job_name = "Harman-tuningjob" + strftime("%d-%H-%M-%S", gmtime())

print(tuning_job_name)

tuning_job_config = {
    "ParameterRanges": {
        "CategoricalParameterRanges": [],
        "ContinuousParameterRanges": [
            {
                "MaxValue": "1",
                "MinValue": "0",
                "Name": "eta",
            },
            {
                "MaxValue": "10",
                "MinValue": "1",
                "Name": "min_child_weight",
            },
            {
                "MaxValue": "2",
                "MinValue": "0",
                "Name": "alpha",
            },
        ],
        "IntegerParameterRanges": [
            {
                "MaxValue": "10",
                "MinValue": "1",
                "Name": "max_depth",
            }
        ],
    },
    "ResourceLimits": {"MaxNumberOfTrainingJobs": 10, "MaxParallelTrainingJobs": 3}
    "Strategy": "Bayesian",
    "HyperParameterTuningJobObjective": {
        "MetricName": "validation:rmse",
        "Type": "Minimize",
    },
}
```

```
Harman-tuningjob06-00-29-25
```

In [31]:

```
output_path = 's3://diabetes-hb/output/'
```

In [32]:

```
#Show output path and training job definition
training_job_definition = {
    "AlgorithmSpecification": {"TrainingImage": container, "TrainingInputMode": "Fi
    "InputDataConfig": [
        {
            "ChannelName": "train",
            "CompressionType": "None",
            "ContentType": "csv",
            "DataSource": {
                "S3DataSource": {
                    "S3DataDistributionType": "FullyReplicated",
                    "S3DataType": "S3Prefix",
                    "S3Uri": training_s3_input_location,
                }
            },
        },
        {
            "ChannelName": "validation",
            "CompressionType": "None",
            "ContentType": "csv",
            "DataSource": {
                "S3DataSource": {
                    "S3DataDistributionType": "FullyReplicated",
                    "S3DataType": "S3Prefix",
                    "S3Uri": validation_s3_input_location,
                }
            },
        },
    ],
    "OutputDataConfig": {"S3OutputPath":output_path},
    "ResourceConfig": {"InstanceCount": 1, "InstanceType": "ml.m4.xlarge", "VolumeS
    "RoleArn": role,
    "StaticHyperParameters": {
        "eval_metric": "rmse",
        "num_round": "100",
        "objective": "reg:squarederror",
    },
    "StoppingCondition": {"MaxRuntimeInSeconds": 43200},
}
```

In [33]:

```
# Launch the hyperparameter tuning job
smclient.create_hyper_parameter_tuning_job(
    HyperParameterTuningJobName=tuning_job_name,
    HyperParameterTuningJobConfig=tuning_job_config,
    TrainingJobDefinition=training_job_definition,
)
```

Out[33]:

```
{'HyperParameterTuningJobArn': 'arn:aws:sagemaker:us-east-1:993566471038:hyper-par
ameter-tuning-job/Harman-tuningjob06-00-29-25',
 'ResponseMetadata': {'RequestId': '679313db-a0a9-4d4e-9e6f-bd3b90323c9d',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': '679313db-a0a9-4d4e-9e6f-bd3b90323c9d',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '128',
   'date': 'Fri, 06 Dec 2024 00:29:26 GMT'},
  'RetryAttempts': 0}}
```

In [44]:

```
smclient.describe_hyper_parameter_tuning_job(HyperParameterTuningJobName=tuning_job
    "HyperParameterTuningJobStatus"
]
```

Out[44]:

```
'Completed'
```

In [45]:

```
smclient.describe_hyper_parameter_tuning_job(HyperParameterTuningJobName=tuning_job
```

Out[45]:

```
'Harman-tuningjob06-00-29-25'
```

In [46]:

```python
tuning_job_result = smclient.describe_hyper_parameter_tuning_job(
    HyperParameterTuningJobName=tuning_job_name
)

status = tuning_job_result["HyperParameterTuningJobStatus"]
if status != "Completed":
    print("Reminder: the tuning job has not been completed.")

job_count = tuning_job_result["TrainingJobStatusCounters"]["Completed"]
print("%d training jobs have completed" % job_count)

objective = tuning_job_result["HyperParameterTuningJobConfig"]["HyperParameterTunin
is_minimize = objective["Type"] != "Maximize"
objective_name = objective["MetricName"]
```

```
10 training jobs have completed
```

In [47]:

```python
from pprint import pprint

if tuning_job_result.get("BestTrainingJob", None):
    print("Best model found so far:")
    pprint(tuning_job_result["BestTrainingJob"])
else:
    print("No training jobs have reported results yet.")
```

```
Best model found so far:
{'CreationTime': datetime.datetime(2024, 12, 6, 0, 33, 57, tzinfo=tzlocal()),
 'FinalHyperParameterTuningJobObjectiveMetric': {'MetricName': 'validation:rmse',
                                                 'Value': 2.4464099407196045},
 'ObjectiveStatus': 'Succeeded',
 'TrainingEndTime': datetime.datetime(2024, 12, 6, 0, 34, 40, tzinfo=tzlocal()),
 'TrainingJobArn': 'arn:aws:sagemaker:us-east-1:993566471038:training-job/Harman-t
uningjob06-00-29-25-008-271e95dd',
 'TrainingJobName': 'Harman-tuningjob06-00-29-25-008-271e95dd',
 'TrainingJobStatus': 'Completed',
 'TrainingStartTime': datetime.datetime(2024, 12, 6, 0, 34, 1, tzinfo=tzlocal()),
 'TunedHyperParameters': {'alpha': '0.9594692971023621',
                          'eta': '0.29362953524634333',
                          'max_depth': '1',
                          'min_child_weight': '4.779271047376941'}}
```

In [48]:

```python
import pandas as pd

tuner = sagemaker.HyperparameterTuningJobAnalytics(tuning_job_name)

full_df = tuner.dataframe()

if len(full_df) > 0:
    df = full_df[full_df["FinalObjectiveValue"] > -float("inf")]
    if len(df) > 0:
        df = df.sort_values("FinalObjectiveValue", ascending=is_minimize)
        print("Number of training jobs with valid objective: %d" % len(df))
        print({"lowest": min(df["FinalObjectiveValue"]), "highest": max(df["FinalOb
        pd.set_option("display.max_colwidth", None)  # Don't truncate TrainingJobNa
    else:
        print("No training jobs have reported valid results yet.")

full_df
```

```
Number of training jobs with valid objective: 10
{'lowest': 2.4464099407196045, 'highest': 3.4649500846862793}
```

Out[48]:

	alpha	eta	max_depth	min_child_weight	TrainingJobName	TrainingJobStatus	FinalObj
0	0.000000	0.163711	8.0	1.000000	Harman-tuningjob06-00-29-25-010-9b015a46	Completed	
1	0.734335	0.313907	4.0	1.000000	Harman-tuningjob06-00-29-25-009-eed70a8b	Completed	
2	0.959469	0.293630	1.0	4.779271	Harman-tuningjob06-00-29-25-008-271e95dd	Completed	
3	0.419534	0.198893	9.0	2.621314	Harman-tuningjob06-00-29-25-007-2ddc2b99	Completed	
4	0.417280	0.335461	5.0	4.676257	Harman-tuningjob06-00-29-25-006-990e9875	Completed	
5	1.166102	0.430364	7.0	3.017242	Harman-tuningjob06-00-29-25-005-14a4ece0	Completed	
6	0.412341	0.339986	4.0	2.615769	Harman-tuningjob06-00-29-25-004-e00d94e0	Completed	
7	0.967255	0.824577	4.0	4.560001	Harman-tuningjob06-00-29-25-003-ad2cc912	Completed	
8	1.797273	0.470682	3.0	1.625163	Harman-tuningjob06-00-29-25-002-f8ee2723	Completed	
9	1.100974	0.990428	6.0	7.756591	Harman-tuningjob06-00-29-25-001-3c47ba80	Completed	

In [49]:

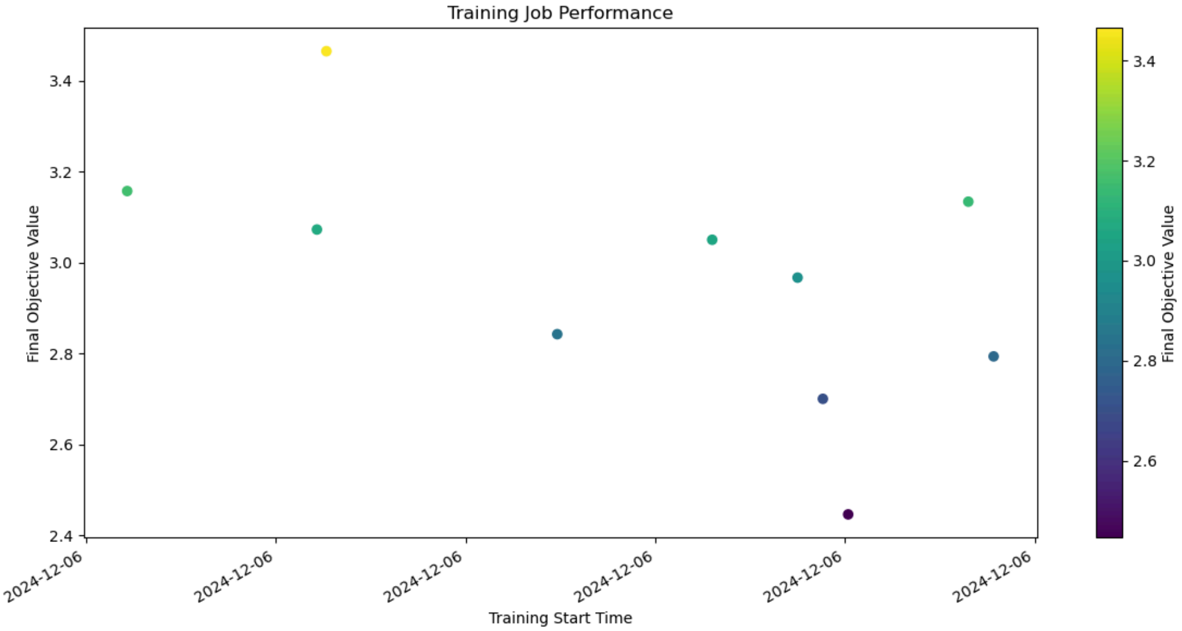
```
full_df.dtypes  # tuner.dataframe() command that we used in previous cell created a
```

Out[49]:

```
alpha                                                float64
eta                                                  float64
max_depth                                            float64
min_child_weight                                     float64
TrainingJobName                                       object
TrainingJobStatus                                     object
FinalObjectiveValue                                  float64
TrainingStartTime              datetime64[ns, tzlocal()]
TrainingEndTime                datetime64[ns, tzlocal()]
TrainingElapsedTimeSeconds                           float64
dtype: object
```

In [50]:

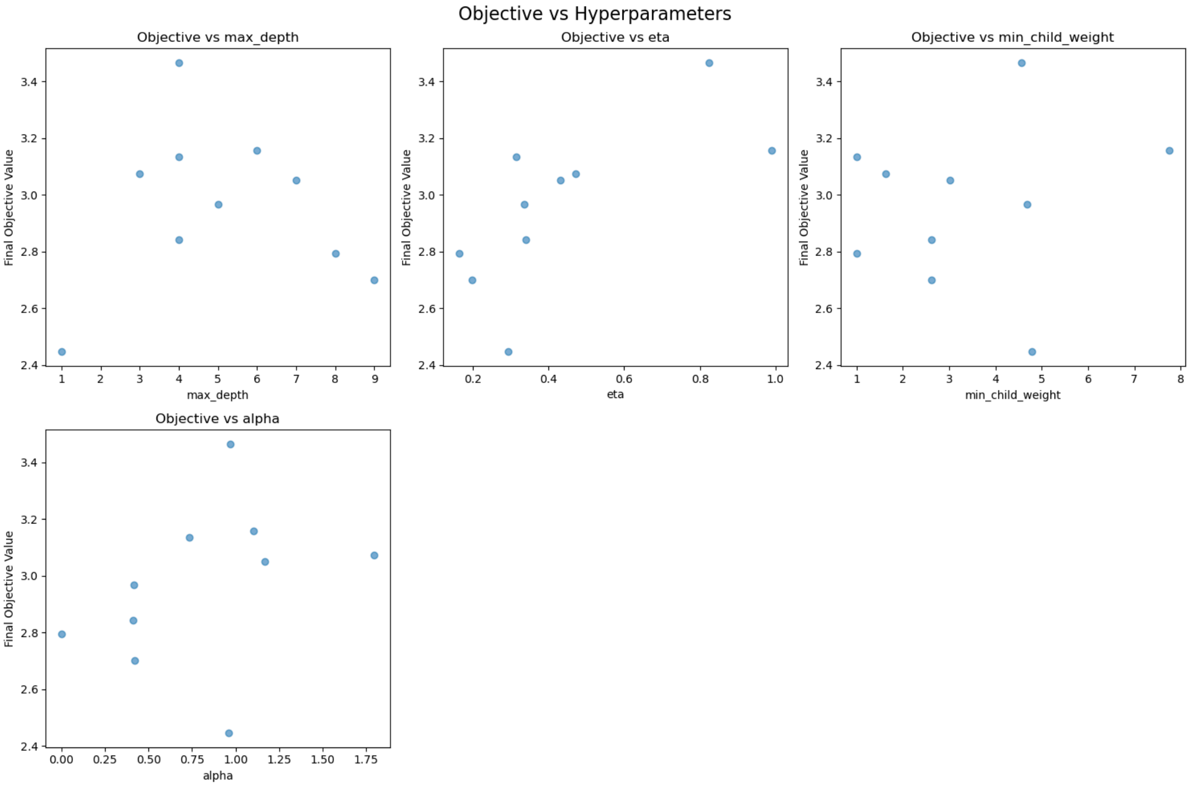
```
import matplotlib.pyplot as plt
import pandas as pd
from matplotlib.dates import DateFormatter
import matplotlib.dates as mdates

# Assuming 'df' is your DataFrame with the data
# If not, you'll need to create it from your data source

# Create the figure and axis
fig, ax = plt.subplots(figsize=(12, 6))

# Plot the data
scatter = ax.scatter(df['TrainingStartTime'], df['FinalObjectiveValue'],
                     c=df['FinalObjectiveValue'], cmap='viridis')

# Format the x-axis to show dates nicely
ax.xaxis.set_major_formatter(DateFormatter('%Y-%m-%d'))
plt.gcf().autofmt_xdate()  # Rotate and align the tick labels

# Set labels and title
ax.set_xlabel('Training Start Time')
ax.set_ylabel('Final Objective Value')
ax.set_title('Training Job Performance')

# Add a colorbar
cbar = plt.colorbar(scatter)
cbar.set_label('Final Objective Value')

# Create the hover annotation
annot = ax.annotate("", xy=(0,0), xytext=(20,20),textcoords="offset points",
                    bbox=dict(boxstyle="round", fc="w"),
                    arrowprops=dict(arrowstyle="->"))
annot.set_visible(False)

def update_annot(ind):
    pos = scatter.get_offsets()[ind["ind"][0]]
    annot.xy = pos
    text = f"TrainingJobName: {full_df['TrainingJobName'].iloc[ind['ind'][0]]}\n"
    text += f"FinalObjectiveValue: {full_df['FinalObjectiveValue'].iloc[ind['ind'][
    for k in tuner.tuning_ranges.keys():
        text += f"{k}: {df[k].iloc[ind['ind'][0]]}\n"
    annot.set_text(text)

def hover(event):
    vis = annot.get_visible()
    if event.inaxes == ax:
        cont, ind = scatter.contains(event)
        if cont:
            update_annot(ind)
            annot.set_visible(True)
            fig.canvas.draw_idle()
        else:
            if vis:
                annot.set_visible(False)
                fig.canvas.draw_idle()

fig.canvas.mpl_connect("motion_notify_event", hover)

plt.tight_layout()
plt.show()
```



In [51]:

```
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.backends.backend_pdf import PdfPages

# Assuming 'df' is your DataFrame with the data
# and 'tuner' is your tuning object with tuning_ranges

def is_num(x):
    try:
        float(x)
        return True
    except ValueError:
        return False

# Create a PDF to save all plots
pdf = PdfPages('hyperparameter_plots.pdf')
```

```
# Get the number of hyperparameters
n_params = len(tuner.tuning_ranges)

# Calculate the number of rows and columns for the subplot grid
n_cols = 3  # You can adjust this
n_rows = (n_params + n_cols - 1) // n_cols

# Create the main figure
fig, axs = plt.subplots(n_rows, n_cols, figsize=(5*n_cols, 5*n_rows))
fig.suptitle(f'Objective vs Hyperparameters', fontsize=16)

# Flatten the axs array for easier iteration
axs = axs.flatten()

for i, (hp_name, hp_range) in enumerate(tuner.tuning_ranges.items()):
    ax = axs[i]

    if hp_range.get("Values"):
        vals = hp_range["Values"]
        if all(is_num(x) for x in vals):
            print(f"Hyperparameter {hp_name} is tuned as categorical, but all value
            x = df[hp_name].astype(float)
        else:
            # For categorical data
            x = df[hp_name]
            ax.set_xticks(range(len(vals)))
            ax.set_xticklabels(vals, rotation=45, ha='right')
    else:
        # For continuous data
        x = df[hp_name]

    # Plot the data
    scatter = ax.scatter(x, df['FinalObjectiveValue'], alpha=0.6)

    # Set labels
    ax.set_xlabel(hp_name)
    ax.set_ylabel('Final Objective Value')
    ax.set_title(f'Objective vs {hp_name}')

    # Add hover functionality
    annot = ax.annotate("", xy=(0,0), xytext=(20,20), textcoords="offset points",
                        bbox=dict(boxstyle="round", fc="w"),
                        arrowprops=dict(arrowstyle="->"))
    annot.set_visible(False)

    def update_annot(ind):
        pos = scatter.get_offsets()[ind["ind"][0]]
        annot.xy = pos
        text = f"{hp_name}: {x.iloc[ind['ind'][0]]}\n"
        text += f"FinalObjectiveValue: {df['FinalObjectiveValue'].iloc[ind['ind'][0
        annot.set_text(text)

    def hover(event):
        vis = annot.get_visible()
        if event.inaxes == ax:
            cont, ind = scatter.contains(event)
            if cont:
                update_annot(ind)
                annot.set_visible(True)
                fig.canvas.draw_idle()
            else:
                if vis:
                    annot.set_visible(False)
                    fig.canvas.draw_idle()

    fig.canvas.mpl_connect("motion_notify_event", hover)

# Remove any unused subplots
for j in range(i+1, len(axs)):
    fig.delaxes(axs[j])

plt.tight_layout()

# Save the figure to the PDF
pdf.savefig(fig)
pdf.close()

# Show the plot
plt.show()
```



In [54]:

```
#Create a new training job using what i think will be the best parameters based on
training_job_name = 'BestJob-final'

training_job_definition = {
    "AlgorithmSpecification": {
        "TrainingImage": container,
        "TrainingInputMode": "File",
    },
    "InputDataConfig": [
        {
            "ChannelName": "train",
            "CompressionType": "None",
            "ContentType": "csv",
            "DataSource": {
                "S3DataSource": {
                    "S3DataDistributionType": "FullyReplicated",
                    "S3DataType": "S3Prefix",
                    "S3Uri": training_s3_input_location,
                }
            },
        },
        {
            "ChannelName": "validation",
            "CompressionType": "None",
            "ContentType": "csv",
            "DataSource": {
                "S3DataSource": {
                    "S3DataDistributionType": "FullyReplicated",
                    "S3DataType": "S3Prefix",
                    "S3Uri": validation_s3_input_location,
                }
            },
        },
    ],
    "OutputDataConfig": {"S3OutputPath": output_path},
    "ResourceConfig": {
        "InstanceCount": 1,
        "InstanceType": "ml.m4.xlarge",
        "VolumeSizeInGB": 10,
    },
    "RoleArn": role,
    "HyperParameters": {
        'objective': 'reg:squarederror',
        'eval_metric': 'rmse',
        'num_round': '100',
        'alpha': '1.2',
        'eta': '0.175',
        'max_depth': '9',
        'min_child_weight': '4.2'
    },
    "StoppingCondition": {"MaxRuntimeInSeconds": 43200},
}
```

In [55]:

```
# Defining what will be in the response variable and running it - this will show al
response= smclient.create_training_job(
    TrainingJobName=training_job_name,
    **training_job_definition
)
response
```

Out[55]:

```
{'TrainingJobArn': 'arn:aws:sagemaker:us-east-1:993566471038:training-job/BestJob-
final',
 'ResponseMetadata': {'RequestId': '0665c31c-8296-43d2-a8e5-8a0c838a222d',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': '0665c31c-8296-43d2-a8e5-8a0c838a222d',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '88',
   'date': 'Fri, 06 Dec 2024 00:38:18 GMT'},
  'RetryAttempts': 0}}
```

In [57]:

```
model_location = 's3://diabetes-hb/output/BestJob/output/model.tar.gz'
```

In [58]:

```
#Create a new model
model_name = "xgboost-serverless" + strftime("%Y-%m-%d-%H-%M-%S", gmtime())
print("Model name: " + model_name)
#Creates environmental variables for the model container
byo_container_env_vars = {"SAGEMAKER_CONTAINER_LOG_LEVEL": "20"}
#Creates the model in sagemaker
create_model_response = smclient.create_model(
    ModelName=model_name,
    Containers=[
        {
            "Image": container,
            "Mode": "SingleModel",
            "ModelDataUrl": model_location,
            "Environment": byo_container_env_vars,
        }
    ],
    ExecutionRoleArn=role,
)
#Shows us where this model is
print("Model Arn: " + create_model_response["ModelArn"])
```

```
Model name: xgboost-serverless2024-12-06-00-39-12
Model Arn: arn:aws:sagemaker:us-east-1:993566471038:model/xgboost-serverless2024-1
2-06-00-39-12
```

In [59]:

```
#Now creating an endpoint configuration to deploy the serverless inference endpoint
#Here is the name of configuration
xgboost_epc_name = "xgboost-serverless-epc" + strftime("%Y-%m-%d-%H-%M-%S", gmtime(
#Create the actual configuration of the endpoint - this will give the endpoint conf
endpoint_config_response =smclient.create_endpoint_config(
    EndpointConfigName=xgboost_epc_name,
    ProductionVariants=[
        {
            "VariantName": "byoVariant",
            "ModelName": model_name,
            "ServerlessConfig": {
                "MemorySizeInMB": 3072,
                "MaxConcurrency": 1,
            },
        },
    ],
)
```

```
print("Endpoint Configuration Arn: " + endpoint_config_response["EndpointConfigArn"
```

```
Endpoint Configuration Arn: arn:aws:sagemaker:us-east-1:993566471038:endpoint-conf
ig/xgboost-serverless-epc2024-12-06-00-39-14
```

In [60]:

```
#This creates the actual endpoint using the endpoint configuration
endpoint_name = "xgboost-serverless-ep" + strftime("%Y-%m-%d-%H-%M-%S", gmtime())

create_endpoint_response = smclient.create_endpoint(
    EndpointName=endpoint_name,
    EndpointConfigName=xgboost_epc_name,
)

print("Endpoint Arn: " + create_endpoint_response["EndpointArn"])
```

```
Endpoint Arn: arn:aws:sagemaker:us-east-1:993566471038:endpoint/xgboost-serverless
-ep2024-12-06-00-39-16
```

In [61]:

```
#This monitors the status of the endpoint while its being created
describe_endpoint_response = smclient.describe_endpoint(EndpointName=endpoint_name)
#This will constantly check to see if the endpoint is creating(every 15 seconds) to
while describe_endpoint_response["EndpointStatus"] == "Creating":
    describe_endpoint_response = smclient.describe_endpoint(EndpointName=endpoint_n
    print(describe_endpoint_response["EndpointStatus"])
    time.sleep(15)

describe_endpoint_response
```

```
Creating
Creating
Creating
Creating
Creating
Creating
Creating
Creating
Creating
Creating
InService
```

Out[61]:

```
{'EndpointName': 'xgboost-serverless-ep2024-12-06-00-39-16',
 'EndpointArn': 'arn:aws:sagemaker:us-east-1:993566471038:endpoint/xgboost-serverl
ess-ep2024-12-06-00-39-16',
 'EndpointConfigName': 'xgboost-serverless-epc2024-12-06-00-39-14',
 'ProductionVariants': [{'VariantName': 'byoVariant',
   'DeployedImages': [{'SpecifiedImage': '683313688378.dkr.ecr.us-east-1.amazonaw
s.com/sagemaker-xgboost:1.5-1',
     'ResolvedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgbo
ost@sha256:c764382b16cd0c921f1b2e66de8684fb999ccbd0c042c95679f0b69bc9cdd12c',
     'ResolutionTime': datetime.datetime(2024, 12, 6, 0, 39, 17, 708000, tzinfo=tz
local())}],
   'CurrentWeight': 1.0,
   'DesiredWeight': 1.0,
   'CurrentInstanceCount': 0,
   'CurrentServerlessConfig': {'MemorySizeInMB': 3072, 'MaxConcurrency': 1}}],
 'EndpointStatus': 'InService',
 'CreationTime': datetime.datetime(2024, 12, 6, 0, 39, 17, 76000, tzinfo=tzlocal
()),
 'LastModifiedTime': datetime.datetime(2024, 12, 6, 0, 41, 46, 527000, tzinfo=tzlo
cal()),
 'ResponseMetadata': {'RequestId': '7b14fecf-9114-4f14-ad40-80399b2d5bef',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': '7b14fecf-9114-4f14-ad40-80399b2d5bef',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '810',
   'date': 'Fri, 06 Dec 2024 00:41:51 GMT'},
  'RetryAttempts': 0}}
```

In [70]:

```python
%%time
runtime = boto3.client('sagemaker-runtime')
#Times how long it takes
#convert the testing data into csv
testing= X_test.to_csv(index=False,header=False)
#invoke the endpoint using the endpoint i created earlier and the testing data
#this will give the raw predictions given by the endpoint
response = runtime.invoke_endpoint(
    EndpointName=endpoint_name,
    Body= testing,
    ContentType="text/csv",
)
#This will help us read the data given by decoding it
import json
y_pred = response['Body'].read().decode('utf-8')
y_pred
```

```
CPU times: user 69.3 ms, sys: 9.62 ms, total: 78.9 ms
Wall time: 274 ms
```

```
Out[70]: '1.327846646308899\n1.327846646308899\n0.7914679646492004\n1.327846646308899\n1.32
         7846646308899\n0.6656466126441956\n1.327846646308899\n1.327846646308899\n1.3278466
         46308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3278466463088
         99\n0.8908554315567017\n0.8994168639183044\n1.327846646308899\n1.327846646308899\n
         0.8901468515396118\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327
         846646308899\n1.1832255125045776\n0.8994168639183044\n1.327846646308899\n1.3278466
         46308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3278466463088
         99\n1.327846646308899\n1.3306552171707153\n1.327846646308899\n1.327846646308899\n
         1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3357
         515335083008\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646
         308899\n1.327846646308899\n1.3306552171707153\n1.327846646308899\n1.32784664630889
         9\n1.327846646308899\n1.3306552171707153\n1.327846646308899\n1.327846646308899\n1.
         327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327846
         646308899\n1.5500707626342773\n1.327846646308899\n1.327846646308899\n1.32784664630
         8899\n1.3306552171707153\n1.327846646308899\n1.327846646308899\n1.327846646308899
         \n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.32
         7846646308899\n0.8994168639183044\n1.327846646308899\n1.327846646308899\n1.4572839
         736938477\n1.327846646308899\n1.327846646308899\n1.327846646308899\n0.899416863918
         3044\n1.327846646308899\n1.0821455717086792\n1.327846646308899\n1.327846646308899
         \n1.3357515335083008\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3
         27846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3278466
         46308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3278466463088
         99\n1.327846646308899\n1.2817890644073486\n1.327846646308899\n1.327846646308899\n
         1.3357515335083008\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327
         846646308899\n1.4923515319824219\n1.327846646308899\n1.327846646308899\n1.32784664
         6308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.41377162933349
         6\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3
         27846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3278466
         46308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3278466463088
         99\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.
         327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n0.890855
         4315567017\n1.3357515335083008\n1.327846646308899\n1.327846646308899\n1.3278466463
         08899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.1832255125045776
         \n1.327846646308899\n1.327846646308899\n0.8994168639183044\n1.327846646308899\n1.3
         27846646308899\n1.413771629333496\n0.8908554315567017\n1.327846646308899\n1.330655
         2171707153\n1.5500707626342773\n1.327846646308899\n1.327846646308899\n1.3278466463
         08899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899
         \n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n0.89
         94168639183044\n1.327846646308899\n1.327846646308899\n1.327846646308899\n0.7914679
         646492004\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.558392763137
         8174\n0.8021594882011414\n1.327846646308899\n1.327846646308899\n1.327846646308899
         \n1.327846646308899\n1.327846646308899\n0.8447977304458618\n1.327846646308899\n1.3
         27846646308899\n1.327846646308899\n1.327846646308899\n1.0821455717086792\n1.327846
         646308899\n1.327846646308899\n0.8994168639183044\n1.327846646308899\n1.32784664630
         8899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.1832255125045776
         \n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.33
         06552171707153\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3278466
         46308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3278466463088
         99\n0.8908554315567017\n1.327846646308899\n1.5583927631378174\n1.327846646308899\n
         1.327846646308899\n1.5583927631378174\n1.1371679306030273\n1.327846646308899\n1.32
         7846646308899\n1.327846646308899\n0.8908554315567017\n1.327846646308899\n1.3278466
         46308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3278466463088
         99\n1.327846646308899\n1.327846646308899\n1.3357515335083008\n1.327846646308899\n
         1.327846646308899\n1.327846646308899\n1.327846646308899\n1.327846646308899\n1.3278
         46646308899\n'
```

In [71]:
```
#Split the prediction into rows and create a dataframe with all the predictions
y_pred = pd.DataFrame(y_pred.split('\n'))
y_pred.columns=['pred']
y_pred.head()
```

Out[71]:

	pred
0	1.327846646308899
1	1.327846646308899
2	0.7914679646492004
3	1.327846646308899
4	1.327846646308899

In [75]:
```
#Check the length of pred and test to make sure they have the same amount of elemen
print(len(y_pred), len(y_test))
```

```
232 231
```

In [76]:
```
# Slice y_pred to match the length of y_test since pred had one more line than test
y_pred = y_pred[:len(y_test)]
```

In [84]:
```
# Since my data is for regression (things like predicting glucose levels, conitnuou
from sklearn.metrics import root_mean_squared_error,mean_absolute_percentage_error
mape = mean_absolute_percentage_error(y_pred,y_test)
rmse = root_mean_squared_error(y_pred,y_test)
print(f'Mape: {mape}\nMSE: {rmse}')
```

```
Mape: 0.7593027987476773
MSE: 1.0768073830921523
```

In [85]:
```
delete_endpoint = smclient.delete_endpoint(EndpointName=endpoint_name)
delete_endpoint
```

Out[85]:
```
{'ResponseMetadata': {'RequestId': 'cc504229-a445-4bcf-b13a-45915bf0de01',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': 'cc504229-a445-4bcf-b13a-45915bf0de01',
   'content-type': 'application/x-amz-json-1.1',
   'date': 'Fri, 06 Dec 2024 00:55:00 GMT',
   'content-length': '0'},
  'RetryAttempts': 0}}
```

In [ ]: