Part A:

1. Create a resource group in your Azure portal and deploy three resources. Azure Data Factory, Azure SQL DB and Blob storage account.

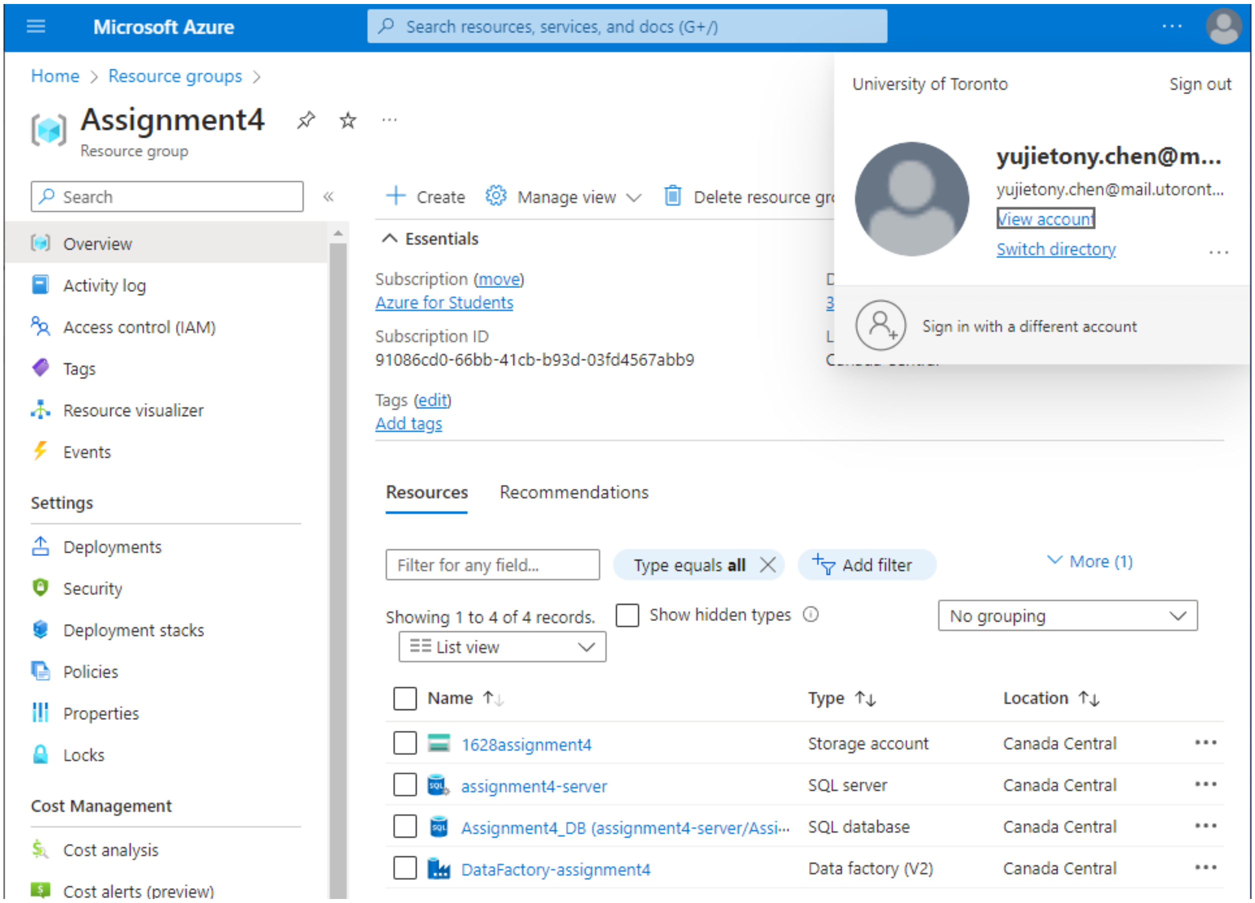


Figure1. List of Azure resource group overview

2. Now create a pipeline in Azure Data Factory and copy gender_jobs_data.csv file from the Blob storage account to Azure SQL DB. (First copy this file from your local machine to Blob Storage).

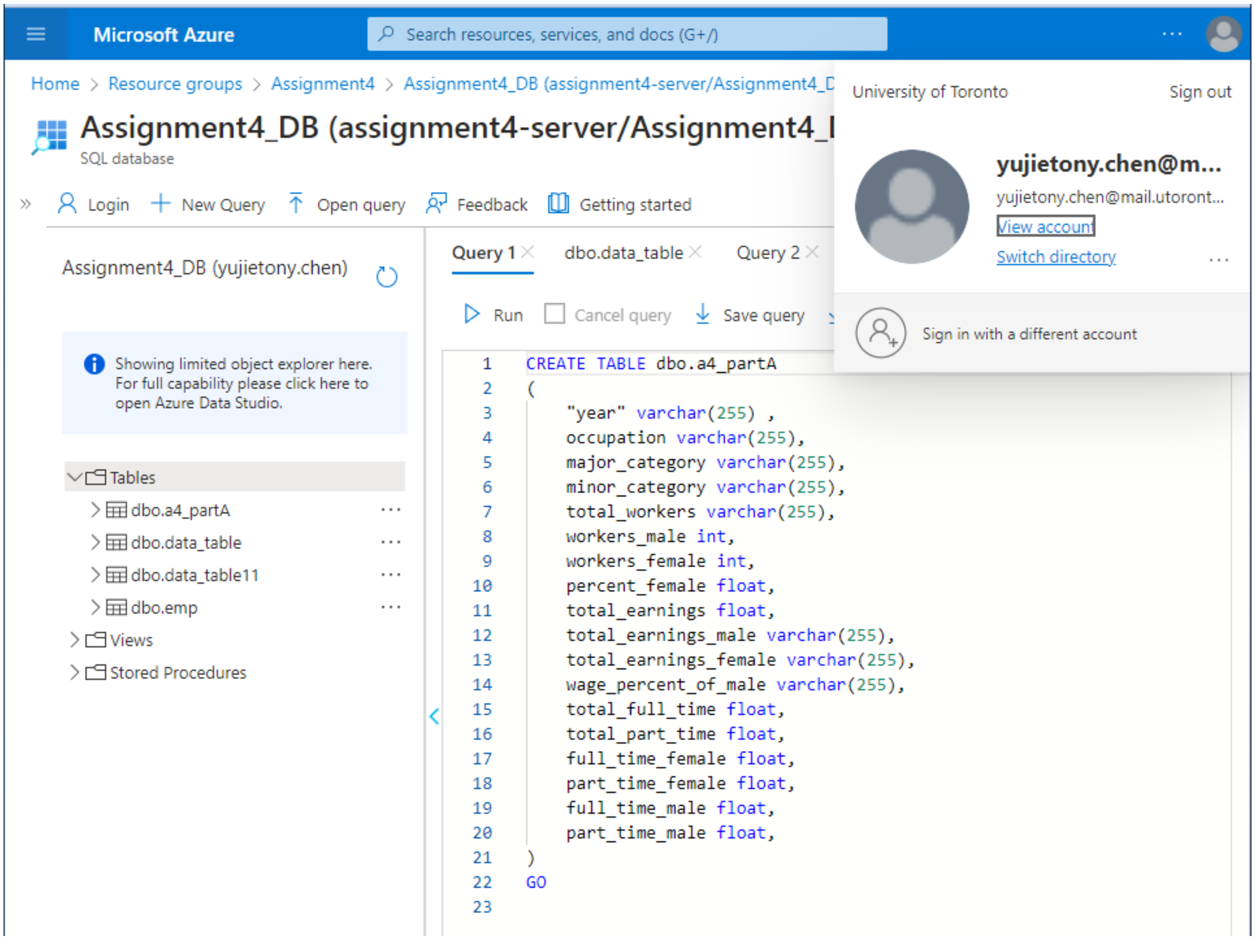


Figure2. Create a data_table in blob storage

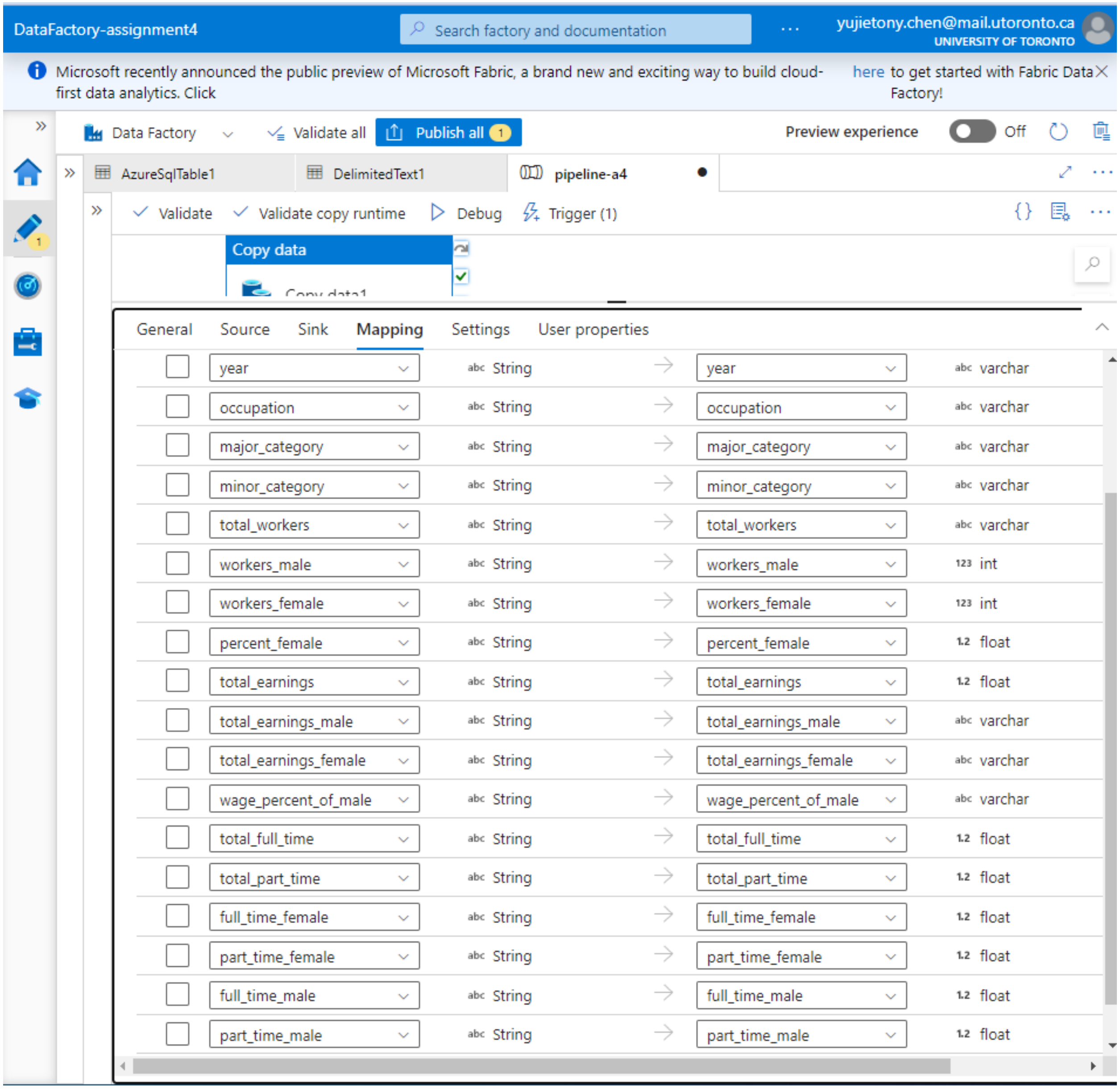


Figure3. Import Schema and specify copy task

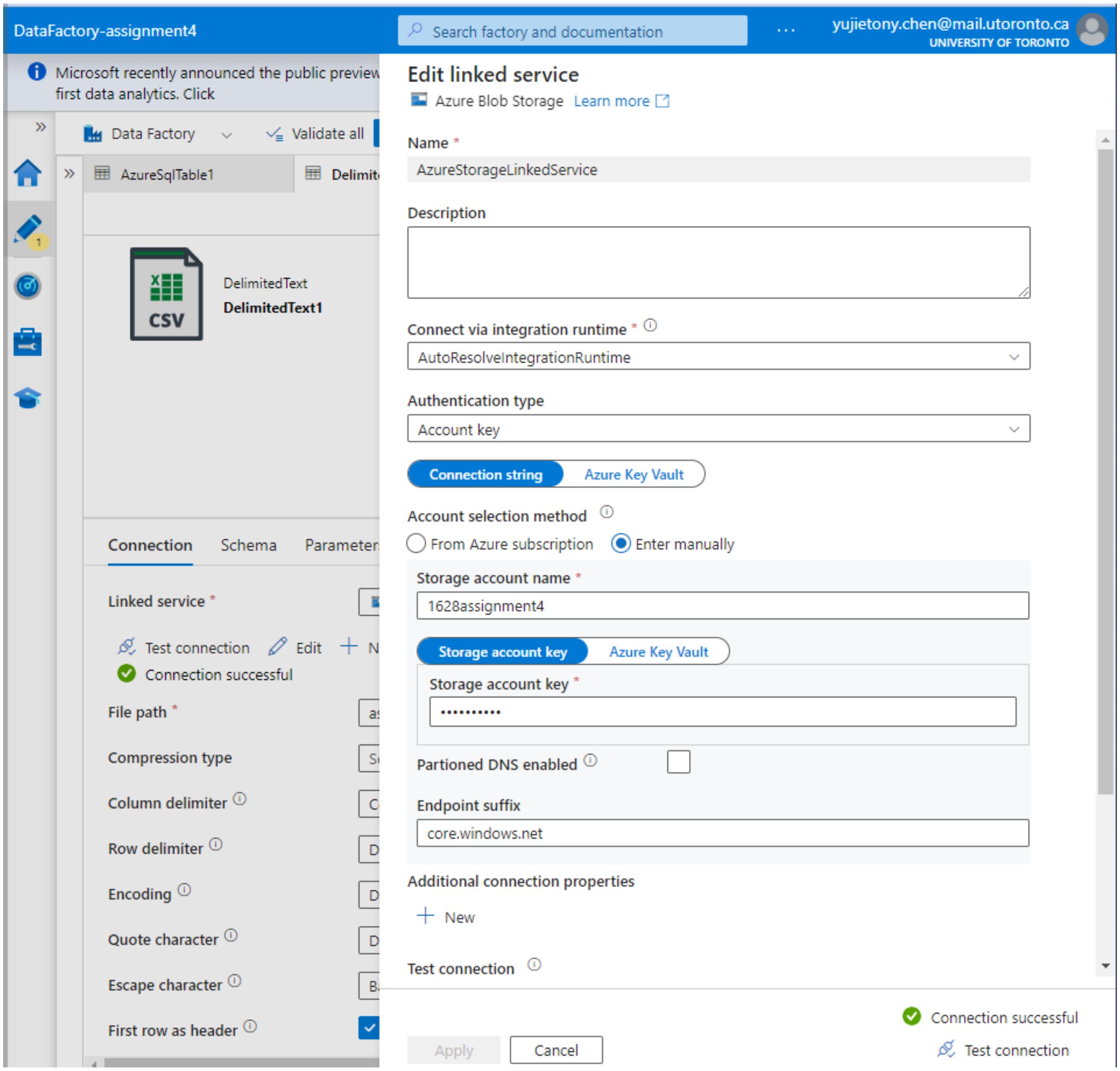


Figure4. Configuration of SourceTable

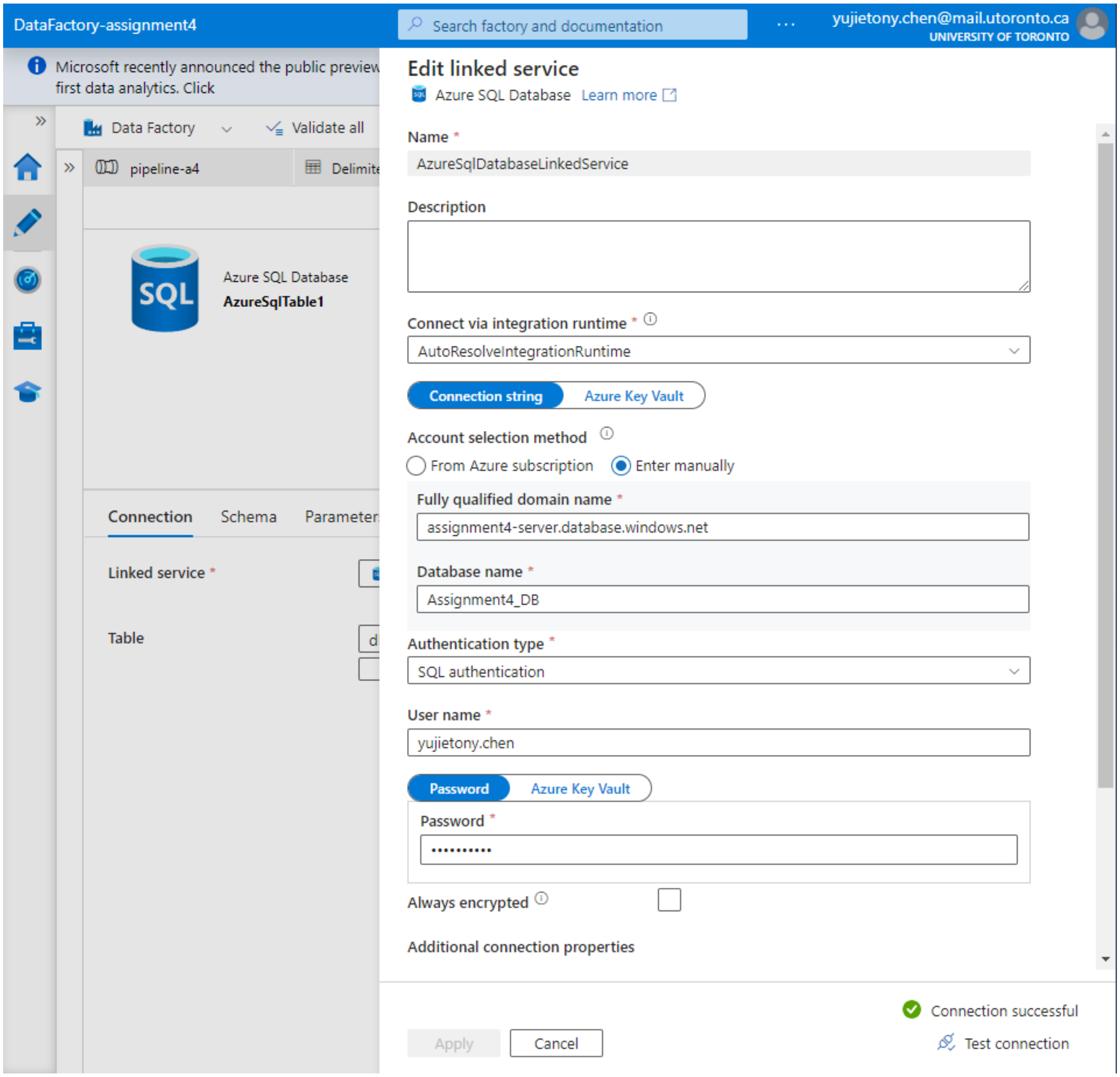


Figure5. Configuration of SinkTable

3. Explain the different types of triggers available in ADF. Now create a schedule trigger and run your pipeline every 3 minutes. Show 5 successful runs.

1. Manual Trigger: Activate the pipeline through user input.
2. Schedule Trigger: schedule a pipeline to run at specific time intervals such as every minute, hour and day.
3. Tumbling Window Trigger: Similar to schedule trigger but it has a built-it time window which allows to define the start and end time of the time interval.
4. Event-based Trigger: Activate the pipeline when a certain event such adding or deleting in blob storage occurs. In addition, logic app can also activate this trigger by calling it.

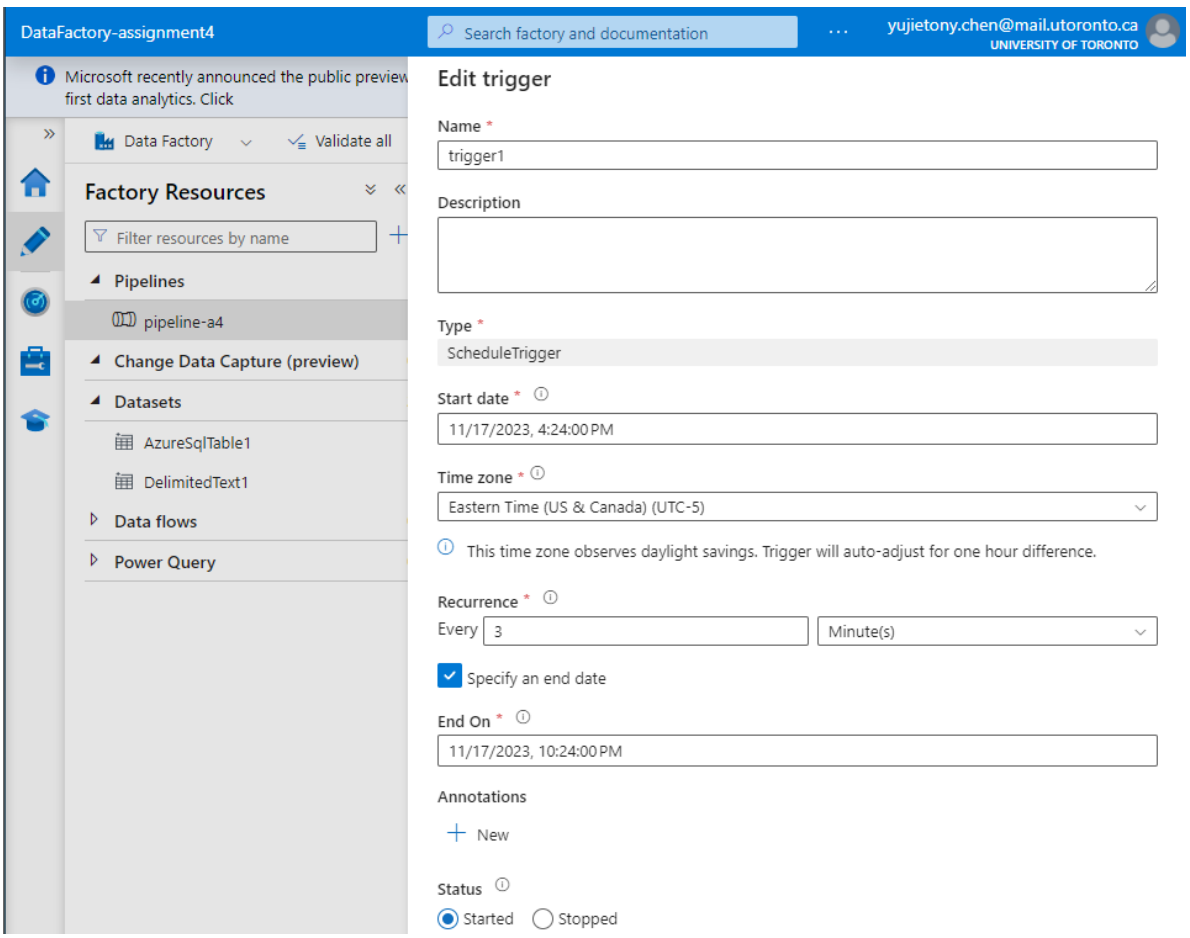


Figure6. Configuration of Trigger

pipeline-a4	11/17/2023, 4:36:00 PM	11/17/2023, 4:36:16 PM	16s	trigger1	Succeeded	Original
pipeline-a4	11/17/2023, 4:33:00 PM	11/17/2023, 4:33:16 PM	16s	trigger1	Succeeded	Original
pipeline-a4	11/17/2023, 4:30:00 PM	11/17/2023, 4:30:16 PM	16s	trigger1	Succeeded	Original
pipeline-a4	11/17/2023, 4:27:00 PM	11/17/2023, 4:27:14 PM	15s	trigger1	Succeeded	Original
pipeline-a4	11/17/2023, 4:24:00 PM	11/17/2023, 4:24:20 PM	20s	trigger1	Succeeded	Original

Figure7. Five successful run of triggers

4. A client needs to replicate objects from ADLS Gen 2 in Canada Central to ADLS Gen 2 in West Europe. Let's say they want to do this in a bi-directional way. How can you set this up

1. Use Azure Data Factory to create linked services for both ADLS Gen2 accounts.
2. Create datasets in ADLS for both accounts and link those datasets in the linked services.
3. Launch Azure Data Factory to create two pipelines that perform copy activities from Canada Central to West Europe and vice versa. Select the correct source and sink tables. For example, pipeline one will copy from the Canada Central source table to the West Europe sink table.
4. In Azure Data Factory, set up two event triggers that will be activated when new blobs are detected in any of the locations. For example, event trigger 1 will activate pipeline 1 when a new blob appears in Canada Central storage.
5. Use Azure Data Factory Monitor to track event status and report any failed activities

PART B:

1. In the gender_jobs_data table - Filter all the OCCUPATIONS in MAJOR_CATEGORY of Computer, Engineering, and Science for the YEAR 2013

Figure8. Code for PartB-1

Results Messages

Search to filter items...

occupation
Actuaries
Aerospace engineers
Agricultural and food science technicians
Agricultural and food scientists
Agricultural engineers
Architects, except naval
Astronomers and physicists
Atmospheric and space scientists
Biological scientists
Biological technicians
Biomedical engineers
Chemical engineers
Chemical technicians
Chemists and materials scientists
Civil engineers
Computer , all other

Figure9. Partial result for PartB-1 (Complete Result will be attach in zip file)

2. In the gender_jobs_data table - How many OCCUPATIONS exist in the MINOR_CATEGORY of Business and Financial Operations overall?

Figure10. Code and result for PartB-2

3. In the gender_jobs_data table - Get all relevant information for bus drivers across all years

Query 1 Query 3 Query 4 **Query 6** Query 7

Run | Cancel query | Save query | Export data as | Show only Editor

```sql
SELECT * FROM [dbo].[a4_partA]
where occupation = 'bus drivers'
```

Results | Messages

Search to filter items...

year	occupation	major_category
2013	Bus drivers	Production, Transportation,
2014	Bus drivers	Production, Transportation,
2015	Bus drivers	Production, Transportation,
2016	Bus drivers	Production, Transportation,
2013	Bus drivers	Production, Transportation,
2014	Bus drivers	Production, Transportation,
2015	Bus drivers	Production, Transportation,

Figure11. Code and partial result for PartB-3 (Complete Result will be attach in zip file)

4. In the gender_jobs_data table - Summarize the total number of WORKERS_FEMALE in the MAJOR_CATEGORY of Management, Business, and Financial by each year.

Run Cancel query Save query Export data as Show only Editor

```sql
SELECT SUM(workers_female) AS total_female_worker,"year"
FROM [dbo].[a4_partA]
WHERE major_category = 'Management, Business, and Financial'
GROUP BY "year"
```

Results Messages

Search to filter items...

total_female_worker	year
7748347	2013
8061480	2014
8381812	2015
8617853	2016

Figure12. Code and result for PartB-4

5. In the gender_jobs_data table - What were the total earnings of male (TOTAL_EARNINGS_MALE) employees in the Service MAJOR_CATEGORY for the year 2015?

Run Cancel query Save query Export data as Show only Editor

```sql
SELECT SUM(TRY_CAST(total_earnings_male As int)) AS total_earning_male_2015
FROM [dbo].[a4_partA]
WHERE major_category = 'Service' AND "year" = '2015'
```

Results Messages

Search to filter items...

total_earning_male_2015
2502426

Figure13. Code and result for PartB-5

6. In the gender_jobs_data table - How many female workers were in management roles in the year 2015?

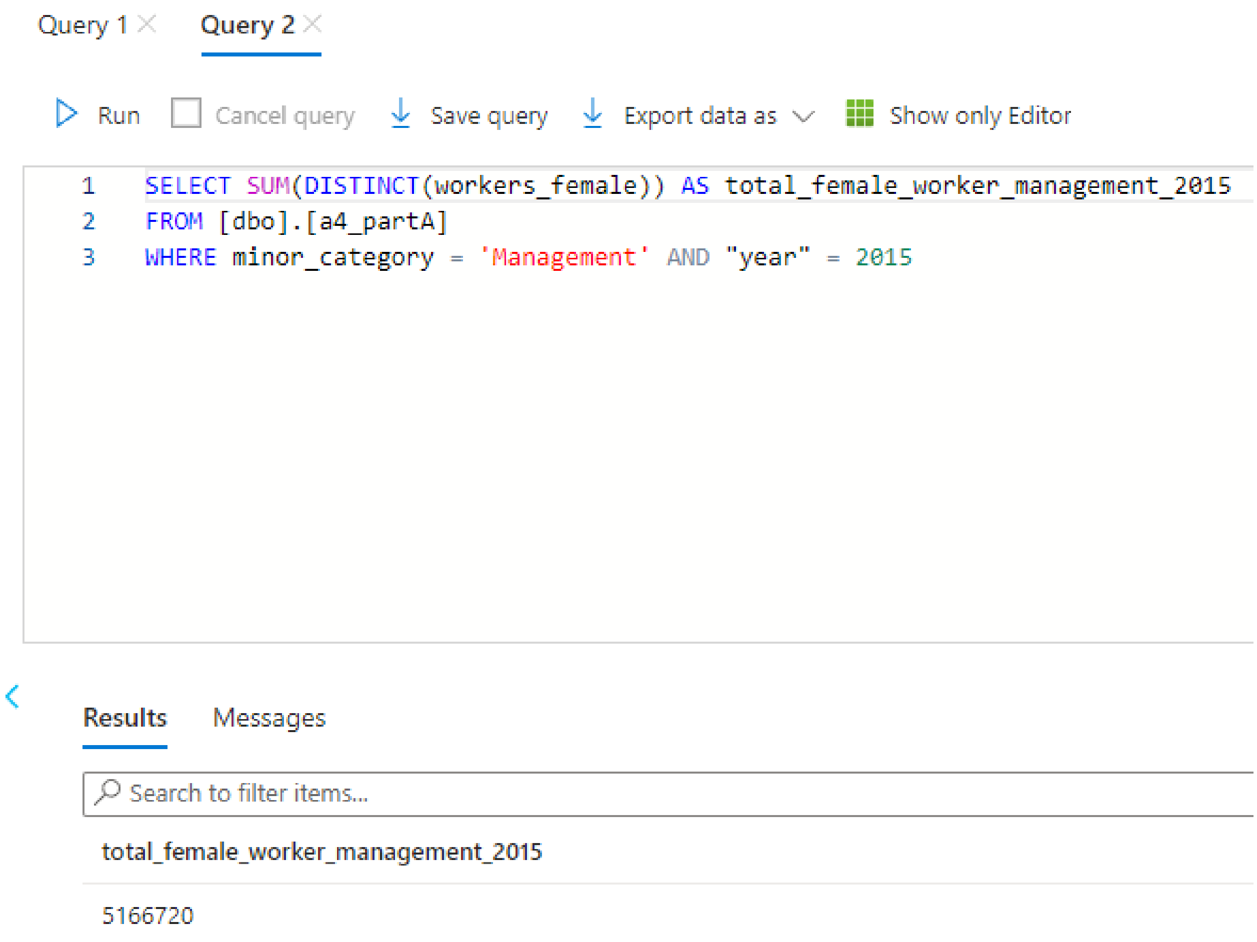


Figure14. Code and result for PartB-6

7. In the gender_jobs_data table - Compare the TOTAL_EARNINGS_MALE and TOTAL_EARNINGS_FEMALE earnings irrespective of occupation by each year

Query 1 Query 3

Run Cancel query Save query Export data as Show only Editor

```sql
SELECT
SUM(TRY_CAST(total_earnings_female As float)) AS total_earnings_female_2015,
SUM(TRY_CAST(total_earnings_male As float)) AS total_earnings_male_2015,"year"
FROM [dbo].[a4_partA]
GROUP BY "year"
```

Results Messages

Search to filter items...

total_earnings_female_2015	total_earnings_male_2015	year
22768521	27754851	2015
22491208	27470450	2014
22054404	27050782	2013
23075602	28463638	2016

Figure15. Code and result for PartB-7

8. In the gender_jobs_data table - How much money (TOTAL_EARNINGS_FEMALE) did female workers make as engineers in 2016?

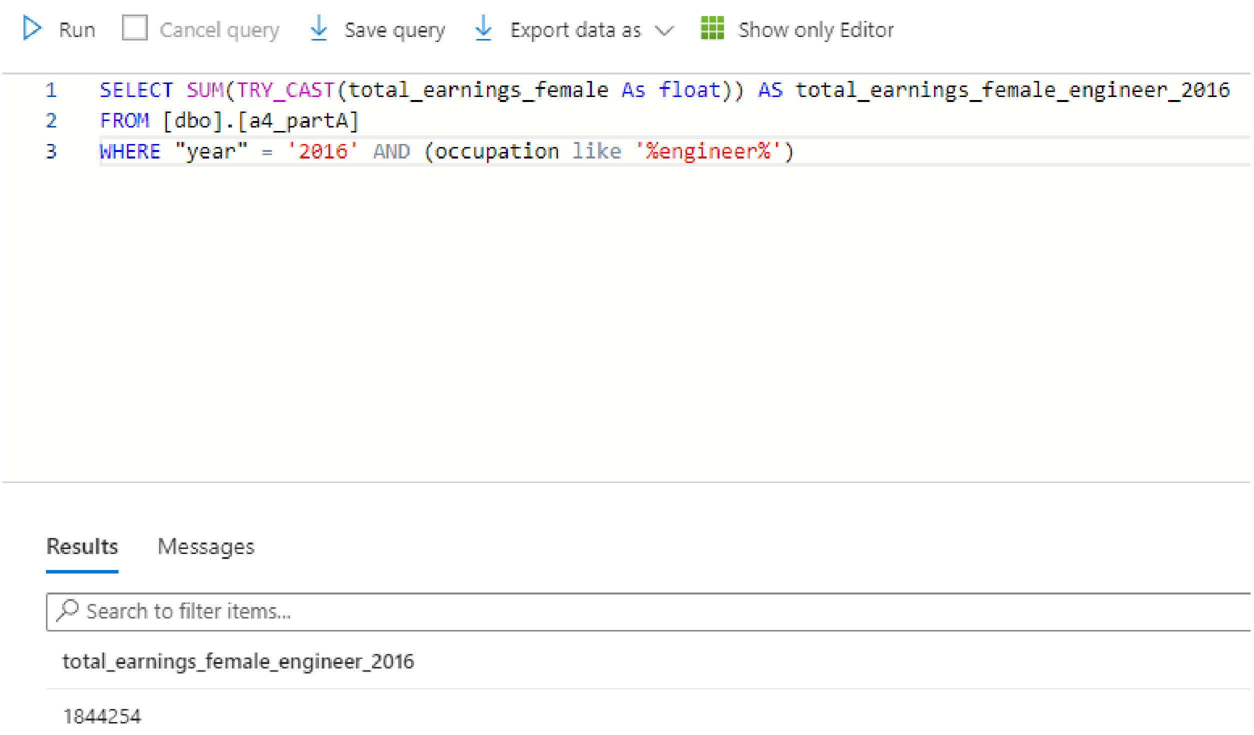


Figure16. Code and result for PartB-8

9. What is the total number of full-time and part-time female workers versus male workers year over year?

Run Cancel query Save query Export data as Show only Editor

```
SELECT
SUM(workers_male * full_time_male/100) AS total_full_time_male,
SUM(workers_female * full_time_female/100) AS total_full_time_female,
SUM(workers_male * part_time_male/100) AS total_part_time_male,
SUM(workers_female * part_time_female/100) AS total_part_time_female,
"year"
FROM [dbo].[a4_partA]
GROUP BY "year"
```

Results Messages

Search to filter items...

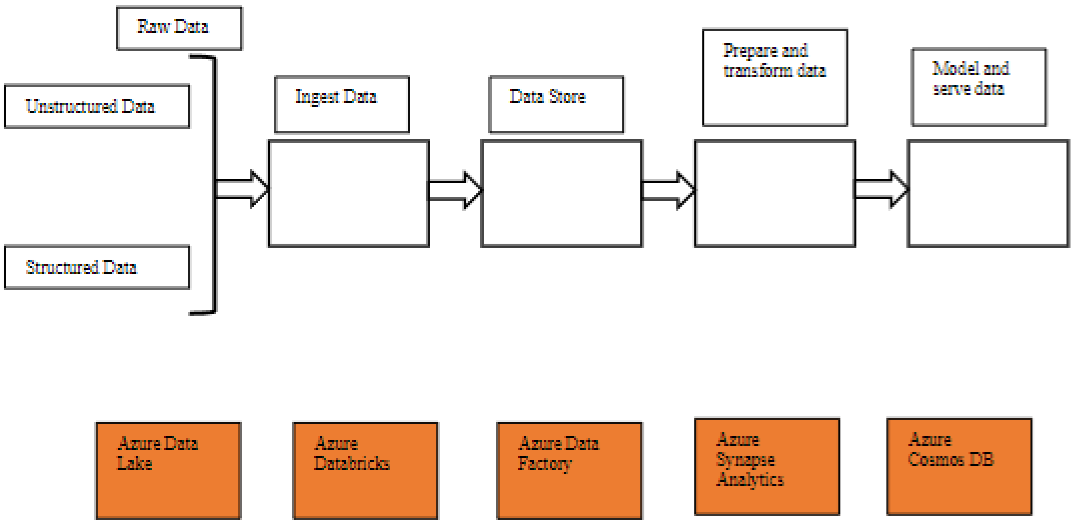
total_full_time_male	total_full_time_female	total_part_time_male	total_part_time_female	year
51720573	33414427.86	7321177	11257267.14	2015
50330271.951	32313480.43	7321815.049	11235684.57	2014
48827487.577	31568143.22	7360645.423	11091509.78	2013
52526792.592	34274127.486	7435299.408	11363858.514	2016

Figure17. Code and result for PartB-9

Second Project:

PART A:

1. Explain below the 5 components shown in orange boxes. Explain which Azure components you will use in this big data architecture and why.



Explanation on components:

1.Azure Data Lake:

A data warehouse that stores and transfer big data. It is specifically designed for large scale data and extreme velocity of data transition. All the data in the data lake are well formatted for analysis tasks.

2.Azure Databricks:

A data Analytics platform that can perform massive scale data engineering and processing.

3.Azure Data Factory:

A data workshop that can create data-driven workflow and pipelines for various activity in the databases. It also provides the ability to integrate and process the raw data into desired forms. Additionally, it can create auto-trigger to activate workflow or pipelines at different time basis.

4.Azure Synapse Analytics:

A fast process engine that is specifically designed for big data takes. It integrates the ability of data warehouse and data analytics for the user to process, manage and analyze data at one place.

5.Azure Cosmos DB:

A serverless database for high performance applications and NoSQL data. It is known for high adoptability and low latency. It is also used all over the world on targeting non-relational data tasks.

Usage in the above architecture:

1.Ingest Data:

-Azure Data Factory can be used to transfer the raw source data into the storage blob by creating data workflow and pipelines.

−Azure Synapse Analytics can be used to create synapse pipelines that move and ingest raw data.

2.Data Store:

-Azure Data Lake is a data warehouse that can be used to data storage and data conversion

-Azure Cosmos DB is a data warehouse to store non-relational data

3.Prepare and transform data:

-Azure Data Factory can be used to process the data for further analysis task by creating data workflow and pipelines.

-Azure Data Brick can be used to perform data analysis and data processing in various kind of programming languages

-Azure Synapse Analytics can create synapse pipelines in SQL, and Spark to perform data transformation and preparation.

4.Model and serve data:

-Azure Synapse Analytics can use SQL to run queries to visualize and serve the data.

-Azure Data Brick supports SQL, spark, python and other programming languages to build plots and diagrams for better serve of the data.

2. Explain how Stream Analytics works in Azure.

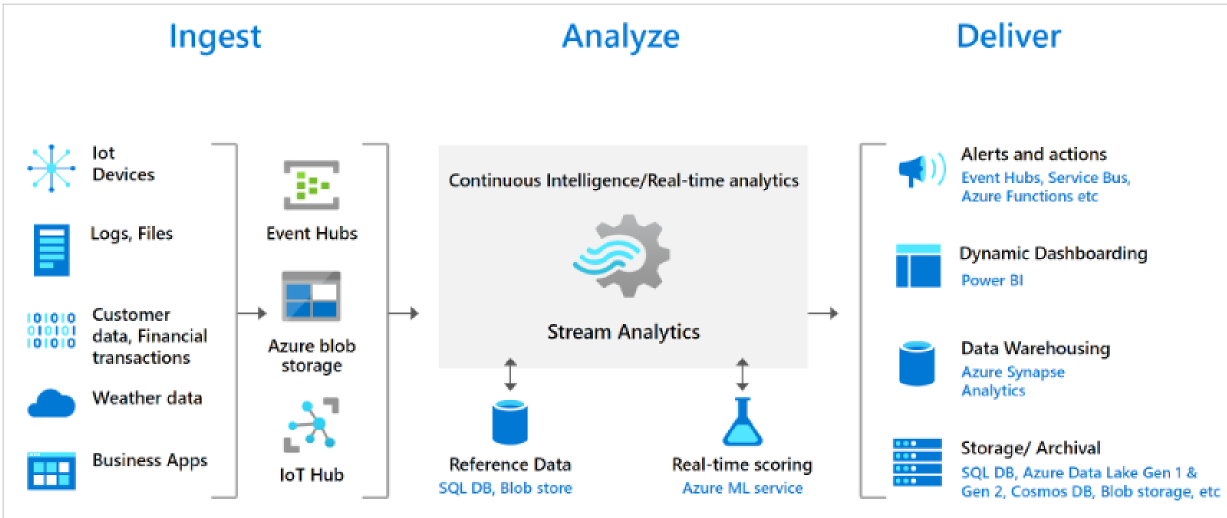


Figure1. Explanation of how stream analytics works in Azure

Azure Stream Analytics operates in three main stages. First comes the ingestion phase, where data is brought in from various sources in real time using Event Hubs, Azure blob storage, or IoT Hub. The next stage involves querying and analyzing the data. Here, Stream Analytics aggregates and analyzes the data through Continuous Intelligence and real-time analytics, incorporating the querying of reference data and real-time scoring. The final stage involves delivering the results, with Azure Stream Analytics capable of outputting the results to dynamic dashboards, data warehouses, storage, or triggering alerts and actions.

3. Deploy all the resources in Azure Portal. Implement a Stream Analytics job by using the Azure portal.

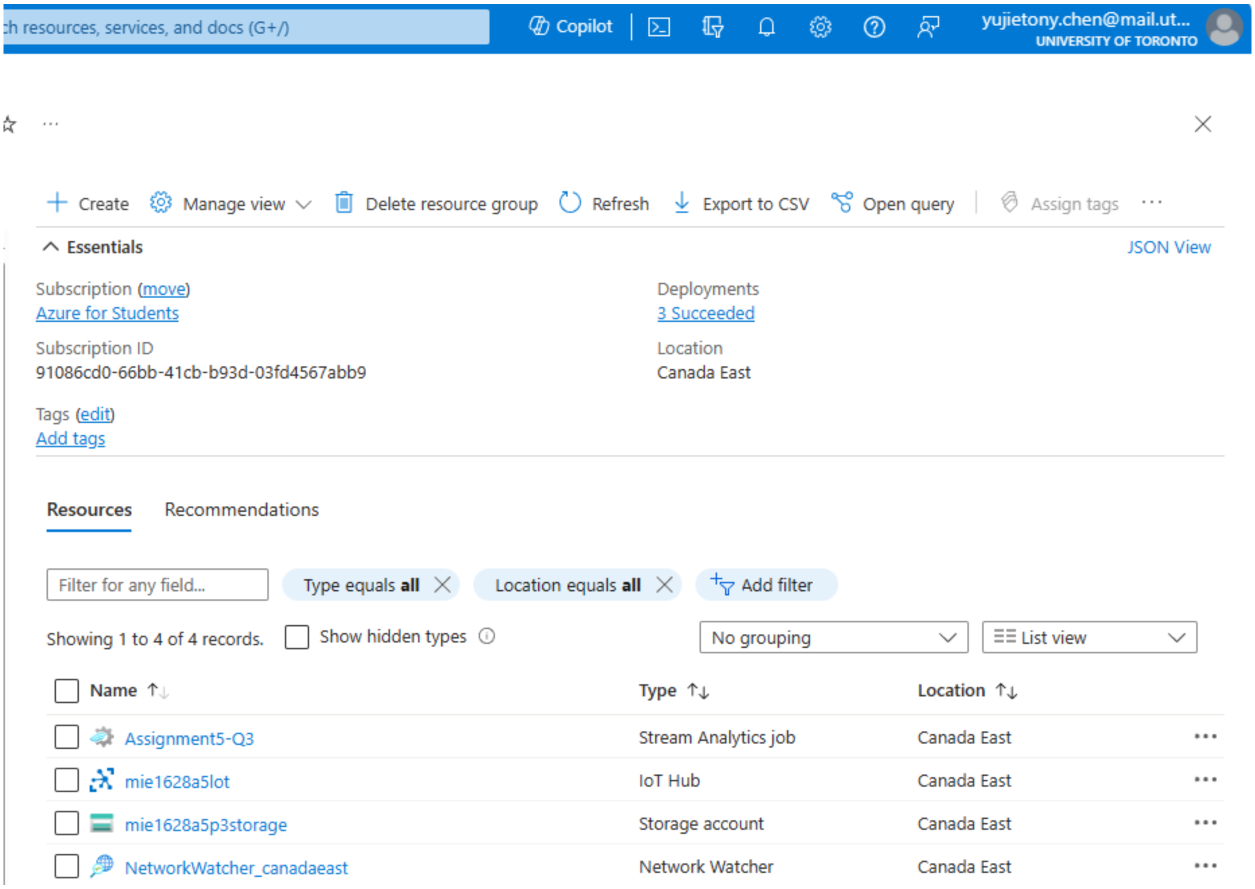


Figure2. Resource deployment in Azure resource group

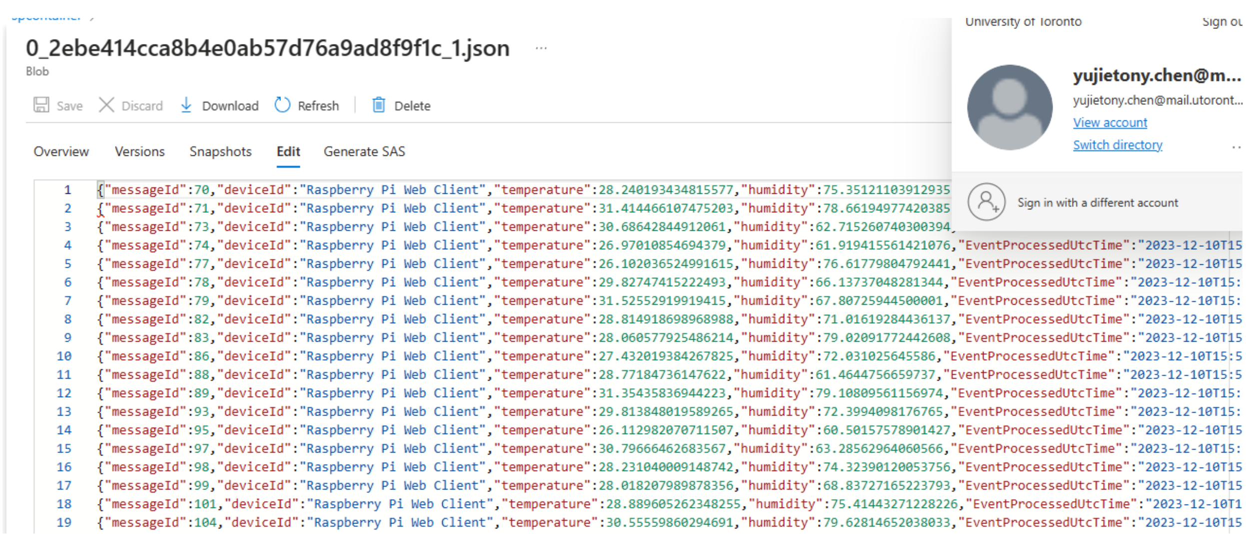


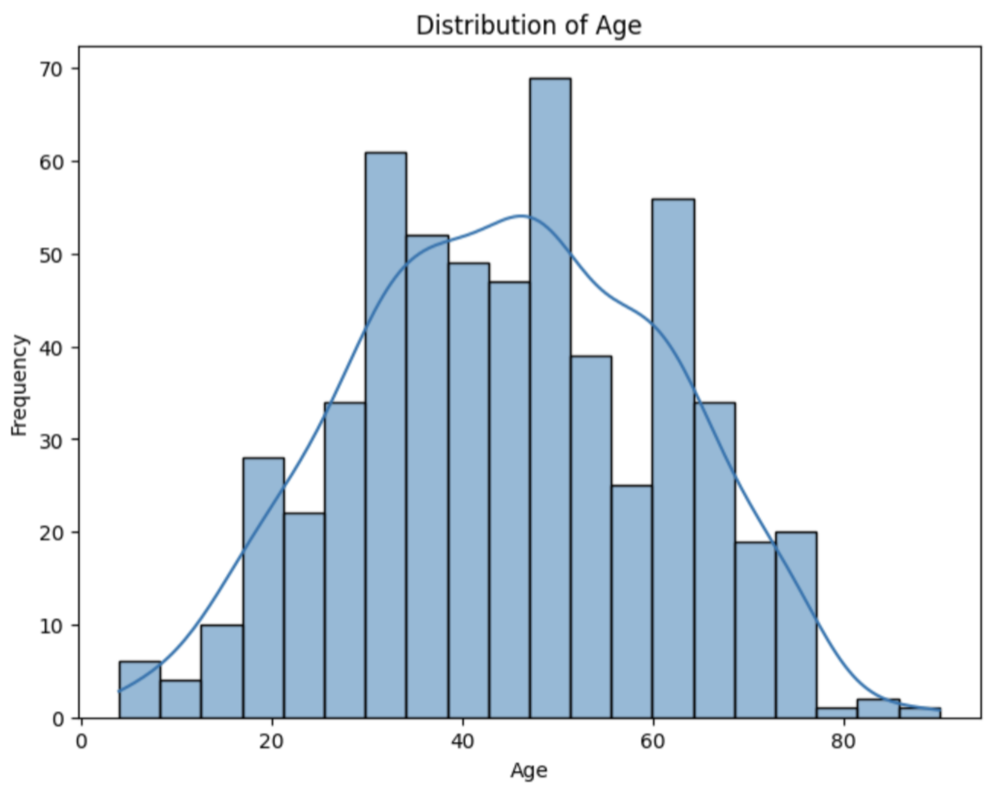
Figure3. IoTHub output from Raspberry Pi device

Part B:
Data Input: ILPD (Indian Liver Patient Dataset)
Link: https://archive.ics.uci.edu/dataset/225/ilpd+indian+liver+patient+dataset

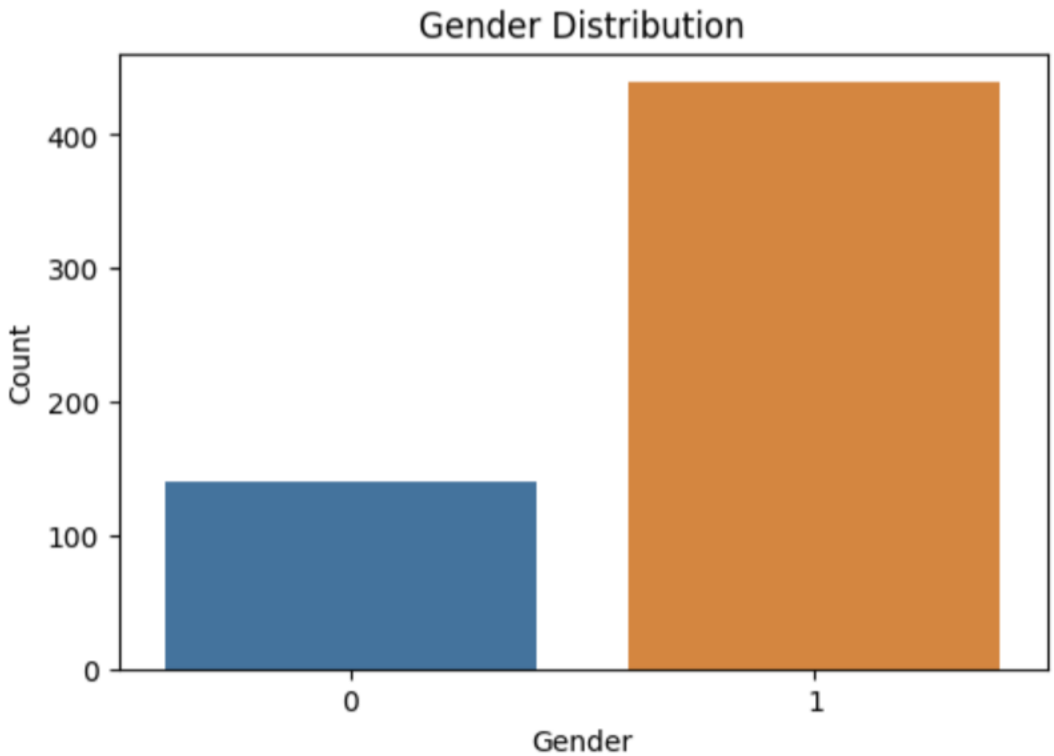
1. Explain what problem you are going to solve using this dataset. Provide a brief overview of your problem statement

We have chosen the ILPD (Indian Patient Dataset) as our source dataset. Our dataset contains 10 features and 583 samples related to the basic information of patients and their levels of different indicators of body status. Our group aims to address a binary classification problem, determining whether the patient does or does not have liver disease. We have selected two models, logistic regression, and random forest, to train our data and will choose the one with the best accuracy. In the end, we will perform hyperparameter tuning on the best model and proofread our results on the test data. The result will be the health assessment of the patient, indicating whether they are affected by liver disease or not.

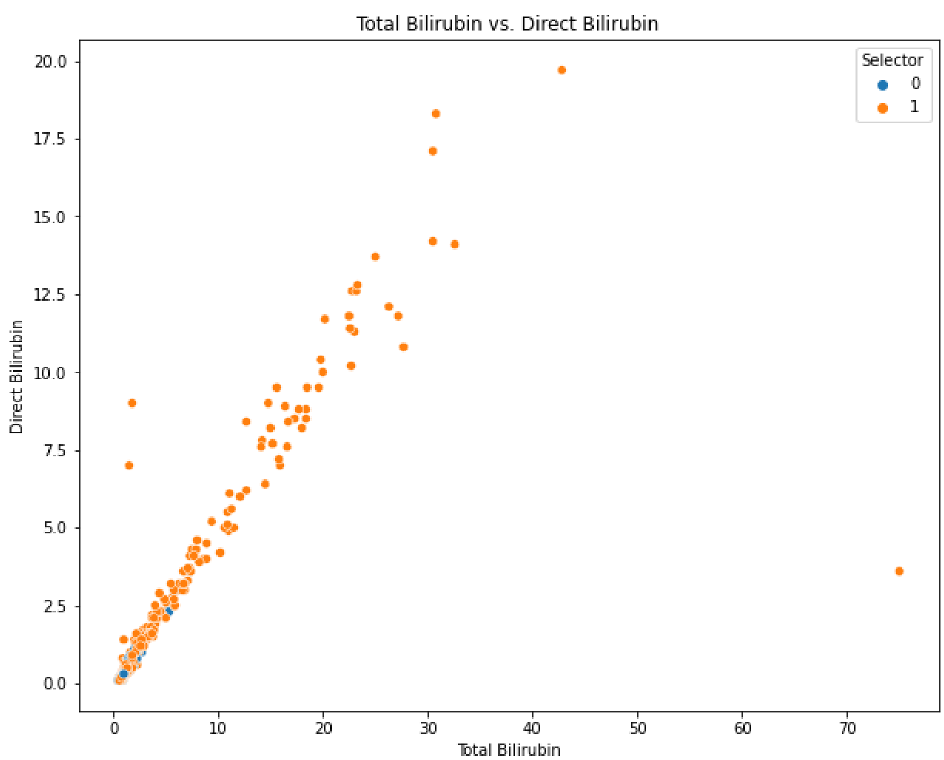
2. Explain your dataset. Explore your dataset and provide at least 5 meaningful charts/graphs with an explanation.



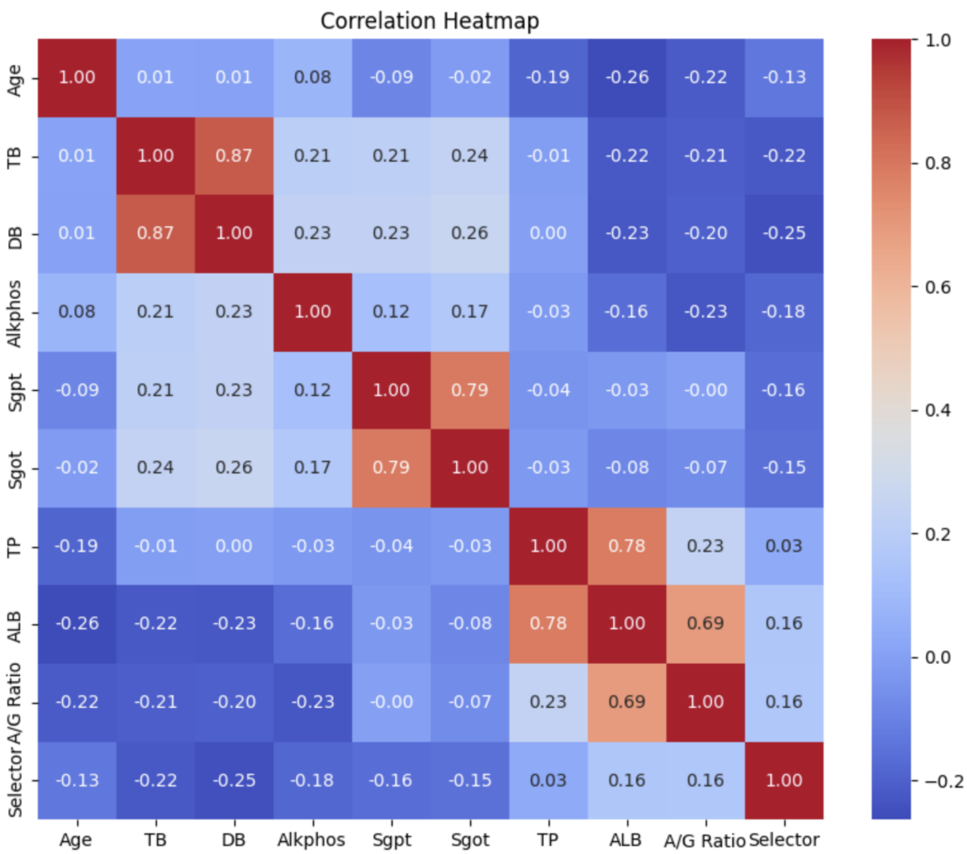
This histogram shows the distribution of ages in the dataset, with the most individuals being around 40 to 50 years old. The distribution appears slightly right-skewed, meaning there are fewer older individuals. The line represents the smoothed probability density of the data.



This bar chart comparing the count of females (0) to males (1) in a dataset. Males significantly outnumber females, as indicated by the height of the bars.



The scatter plot shows the relationship between Total Bilirubin and Direct Bilirubin levels in individuals, with two groups represented: those with liver disease (1) and those without liver disease (0). The plot suggests that as Total Bilirubin levels increase, Direct Bilirubin levels also increase, and this trend is visible in both groups. However, individuals with liver disease (1) may have higher levels of both Total and Direct Bilirubin overall, as indicated by the cluster of orange points higher up on the y-axis.



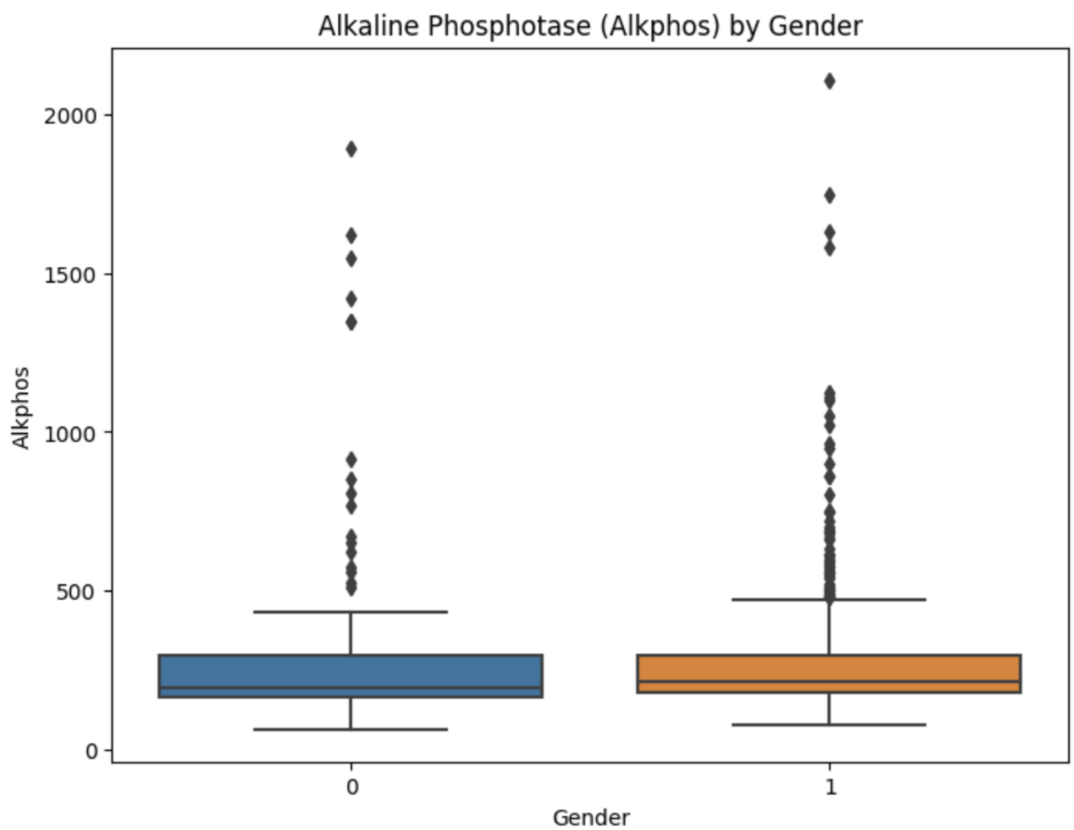
The image shows a correlation heatmap, which is a graphical representation of the correlation matrix between different variables. Each cell shows the correlation coefficient between two variables, ranging from -1 to 1. A correlation of 1 implies a perfect positive relationship, -1 implies a perfect negative relationship, and 0 implies no relationship.

The variables included are Age, Total Bilirubin (TB), Direct Bilirubin (DB), Alkaline Phosphotase (Alkphos), Aspartate Aminotransferase (Sgpt), Alanine Aminotransferase (Sgot), Total Proteins (TP), Albumin (ALB), Albumin/Globulin Ratio (A/G Ratio), and a Selector (indicating with or without liver disease).

There is a strong positive correlation between TB and DB, which is expected since DB is a component of TB.

Sgpt and Sgot also show a strong positive correlation, which is common as they are both liver enzymes.

ALB and A/G Ratio have a strong positive correlation, likely reflecting the fact that albumin is a major component of the total protein that influences the A/G ratio.
Age does not seem to have a strong linear correlation with any of the liver function tests in this dataset.



This is a box plot showing the distribution of Alkaline Phosphatase (Alkphos) levels across two gender categories: female (0) and male (1). The central line in each box represents the median Alkphos level, the box edges represent the interquartile range (IQR), and the whiskers extend to show the rest of the distribution, except for outliers, which are represented as individual points beyond the whiskers.
Both genders have outliers with high Alkphos levels.
The median level of Alkphos is slightly higher for males (1) than females (0).
The spread (variability) of Alkphos levels is greater in males than in females, as indicated by the longer whiskers and larger number of outliers.
The IQR for males is slightly larger than for females, suggesting more variability around the median for males.

3. Do data cleaning/pre-processing as required and explain what you have done for your dataset and why.

Our dataset only contains several missing values, which has been drop from the dataset, and most of the data is pre-processed and presented in numerical form. Secondly, there is only one feature where gender has been expressed as a string, and it can be easily encoded in binary form. Afterward, we will split the dataset into train, valid, and test sets and build a machine-learning model to tackle the problem.

4. Implement 2 machine learning models, explain which algorithms you have selected and why. Compare them and show success metrics (Accuracy/RMSE/Confusion Matrix) as per your problem.

We choose the Logistic Regression and Random Forest. Logistic Regression is chosen for its simplicity and interpret ability, particularly in binary classification problems where the relationship between variables is linear or near-linear. It offers a probabilistic understanding of the model's predictions. On the other hand, Random Forest is selected for its ability to handle complex, non-linear relationships without the need for feature transformation. As an ensemble method, it often yields higher accuracy and robustness against over-fitting compared to individual decision trees, making it suitable for both classification and regression tasks with more complex datasets. Both algorithms provide valuable insights into feature importance, aiding in understanding the underlying patterns within the data.
Since our problem is evaluated based on knowledge of medical fields. The cost of false negative is too great and we should aim the model to achieve a high recall where there should be 0 false negative value.

Logistic Regression Model

```python
# Implementing Logistic Regression
model = LogisticRegression(random_state=0, solver='liblinear',max_iter = 1000).fit(X_train, y_train)

# Obtain prediction
prediction = model.predict(X_test)

accuracy, recall, confusion = evaluate_model(prediction,y_test)

print('Logistic Regression model has %.5f accuracy.' %(accuracy))
print('Logistic Regression model has %.5f recall score.' %(recall))
print('Logistic Regression model confusion matrix shown as.\n', confusion)
```

[92] ✓ <1 sec - Command executed in 411 ms by yujietony.chen on 5:42:31 PM, 12/09/23

```
Logistic Regression model has 0.71839 accuracy.
Logistic Regression model has 0.96721 recall score.
Logistic Regression model confusion matrix shown as.
 [[  7  45]
 [  4 118]]
```

The image shows the output of a logistic regression model with an accuracy of 71.839% and a very high recall of 96.721%. The confusion matrix indicates that the model predicted 118 true positives and 7 true negatives, but it also incorrectly predicted 45 false positives and only 4 false negatives. The high recall score means the model is very good at identifying the positive class.

Random Forest Model

```python
# Implementing Random Forest
model = RandomForestClassifier(max_depth=2, random_state=0).fit(X_train, y_train)

# Obtain prediction
prediction = model.predict(X_test)

accuracy, recall, confusion = evaluate_model(prediction,y_test)

print('Logistic Regression model has %.5f accuracy.' %(accuracy))
print('Logistic Regression model has %.5f recall score.' %(recall))
print('Logistic Regression model confusion matrix shown as.\n', confusion)
```

[91] ✓ 1 sec - Command executed in 1 sec 37 ms by yujietony.chen on 5:39:55 PM, 12/09/23

```
Logistic Regression model has 0.70115 accuracy.
Logistic Regression model has 1.00000 recall score.
Logistic Regression model confusion matrix shown as.
 [[  0  52]
 [  0 122]]
```

The Random Forest model achieved an accuracy of 70.115% and a recall of 100%. The confusion matrix shows that the model predicted 122 instances as positive (true positives) and none as negative, resulting in 52 false negatives and zero true negatives or false positives. This result looks good, since we want a high recall.

5. Do hyperparameter tuning for your algorithms.

We decided to apply hyperparameter tuning for our algorithm. For logistic regression, we selected C and penalty as our tuned hyperparameters. Furthermore, C represents the inverse of regularization strength, and by increasing the value of C, the algorithm tends to focus more on fitting the training data exactly. Next, the penalty defines the choice of the regularization method that we pick. According to our results, the best model achieves a perfect recall score with a C value of 0.001 and L1 regularization.

Logistic Regression

```python
# Define the hyperparameter grid
param_grid = {'C': [0.001, 0.01, 0.1, 1, 10, 100], 'penalty': ['l1', 'l2']}

# Create the logistic regression model
logistic_model = LogisticRegression(random_state=0, solver='liblinear',max_iter = 1000)

# Create GridSearchCV with 5-fold cross-validation
grid_search = GridSearchCV(logistic_model, param_grid, cv=5, scoring='accuracy')

# Fit the model to the training data
grid_search.fit(X_train, y_train)

# Obtain the best hyperparameters
best_params = grid_search.best_params_

# Obtain prediction using the best model
best_model = grid_search.best_estimator_
prediction = best_model.predict(X_test)

accuracy, recall, confusion = evaluate_model(prediction,y_test)

print(f'Best Hyperparameters: {best_params}')
print('Logistic Regression model has %.5f accuracy.' %(accuracy))
print('Logistic Regression model has %.5f recall score.' %(recall))
print('Logistic Regression model confusion matrix shown as.\n', confusion)
```

[93] ✓ 1 sec - Command executed in 1 sec 860 ms by yujietony.chen on 5:44:08 PM, 12/09/23

```
Best Hyperparameters: {'C': 0.001, 'penalty': 'l1'}
Logistic Regression model has 0.70115 accuracy.
Logistic Regression model has 1.00000 recall score.
Logistic Regression model confusion matrix shown as.
 [[  0  52]
 [  0 122]]
```

Afterward, we conducted the same tuning process for the Random Forest with the number of estimators, maximum depth, and minimum samples of split. It turns out that the Random Forest performance is reduced after tuning, with the highest accuracy of 66% and a recall of 86.89%.

Random Forest

```python
# Define the hyperparameter grid
param_grid = {
    'n_estimators': [50, 100, 150],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
}

# Create the Random Forest model
rf_model = RandomForestClassifier(random_state=0)

# Create GridSearchCV with 5-fold cross-validation
grid_search = GridSearchCV(rf_model, param_grid, cv=5, scoring='accuracy')

# Fit the model to the training data
grid_search.fit(X_train, y_train)

# Obtain the best hyperparameters
best_params = grid_search.best_params_

# Obtain prediction using the best model
best_model = grid_search.best_estimator_
prediction = best_model.predict(X_test)

accuracy, recall, confusion = evaluate_model(prediction,y_test)

print(f'Best Hyperparameters: {best_params}')
print('Logistic Regression model has %.5f accuracy.' %(accuracy))
print('Logistic Regression model has %.5f recall score.' %(recall))
print('Logistic Regression model confusion matrix shown as.\n', confusion)
```

[94] 40 sec - Command executed in 40 sec 890 ms by yujietony.chen on 5:44:55 PM, 12/09/23

```
Best Hyperparameters: {'max_depth': 10, 'min_samples_split': 5, 'n_estimators': 50}
Logistic Regression model has 0.67241 accuracy.
Logistic Regression model has 0.86885 recall score.
Logistic Regression model confusion matrix shown as.
 [[ 11  41]
 [ 16 106]]
```