

Predicting per capita income percentile group

Advanced Data Science Deep Learning techniques for informed decision making

By

Senior Statistician Mbaye Kebe

Content

- Business Use Case
- The NHIES 2016 Dataset Used
- Architectural Decisions
- Process Model:
 - ▶ Process Model Task 1 - Initial Data Exploration
 - ▶ Process Model Task 2 - Extract, Transform, Load (ETL)
 - ▶ Process Model Task 3 - Feature Creation/Engineering
 - ▶ Process Model Task 4 - Model Definition
 - ▶ Process Model Task 5 - Model Training
 - ▶ Process Model Task 6 - Model Evaluation
 - ▶ Process Model Task 7 - Model Deployment
- Conclusions

Business Use Case

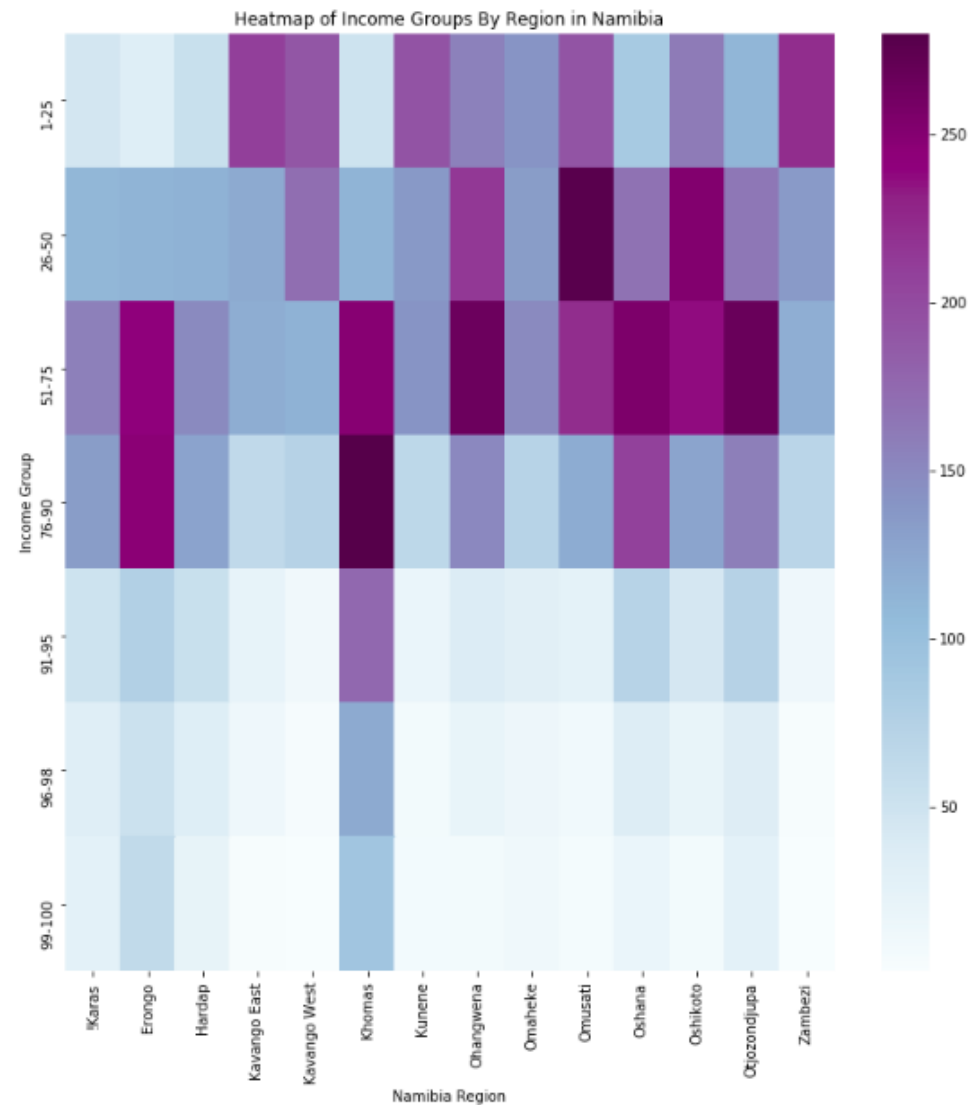
- ▶ In many countries, public decision makers would want to know more about the factors that mostly correlate with household income.
- ▶ Knowing these factors will allow them to put in place programs and projects for poverty alleviation for instance.
- ▶ Development institutions are also interested in being able to look at factors influencing income and classify households into income groups to better address their needs.
- ▶ In both cases, data science techniques with machine learning algorithms can help achieve the objectives in an efficient and cost-effective manner.

The NHIES 2016 Dataset Used

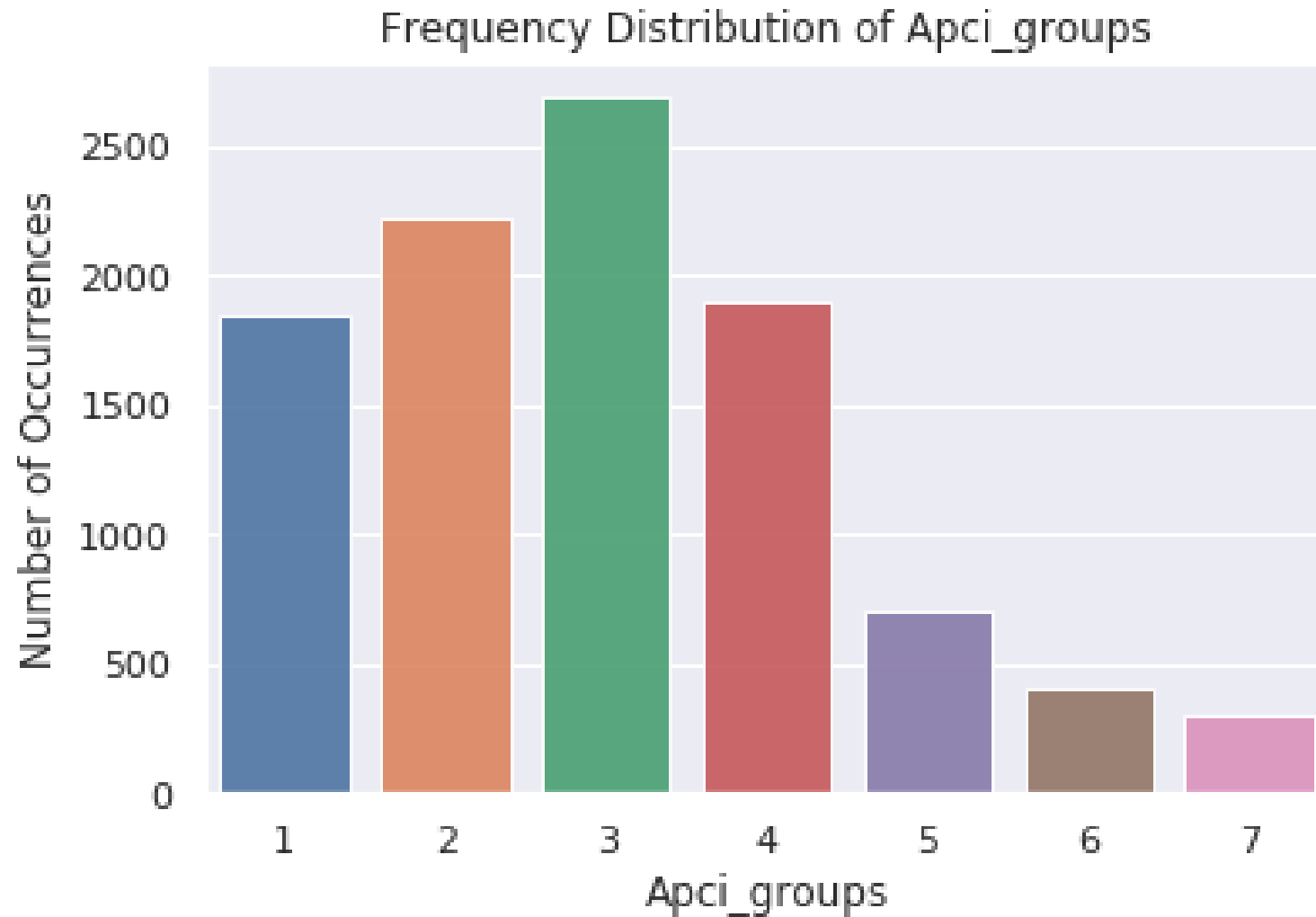
- ▶ The Namibia Household Income/Expenditure Survey of 2015-2016 was used. It is publicly available in an anonymized form at <https://nsa.org.na/page/central-data-catalogue/>
- ▶ There are 10090 rows (or observations, samples) and 2485 columns (or variables many with household income related features).
- ▶ The key dependent variable -label- to predict is the adjusted per capita income percentiles ('apci_groups').
- ▶ In the next slide the Heat Map shows the data with label cross-tabulated against regions. Looking at the map, for instance, one can see that most high income groups are in Khomas region.

```
[12]: import seaborn as sns
fig = plt.figure(figsize=(12,12))
r = sns.heatmap(apci_region, cmap='BuPu')
r.set_title("Heatmap of Income Groups By Region in Namibia")
```

```
[12]: Text(0.5, 1.0, 'Heatmap of Income Groups By Region in Namibia')
```



Apci_groups bar chart



1:1-25

2:26-50

3:51-75

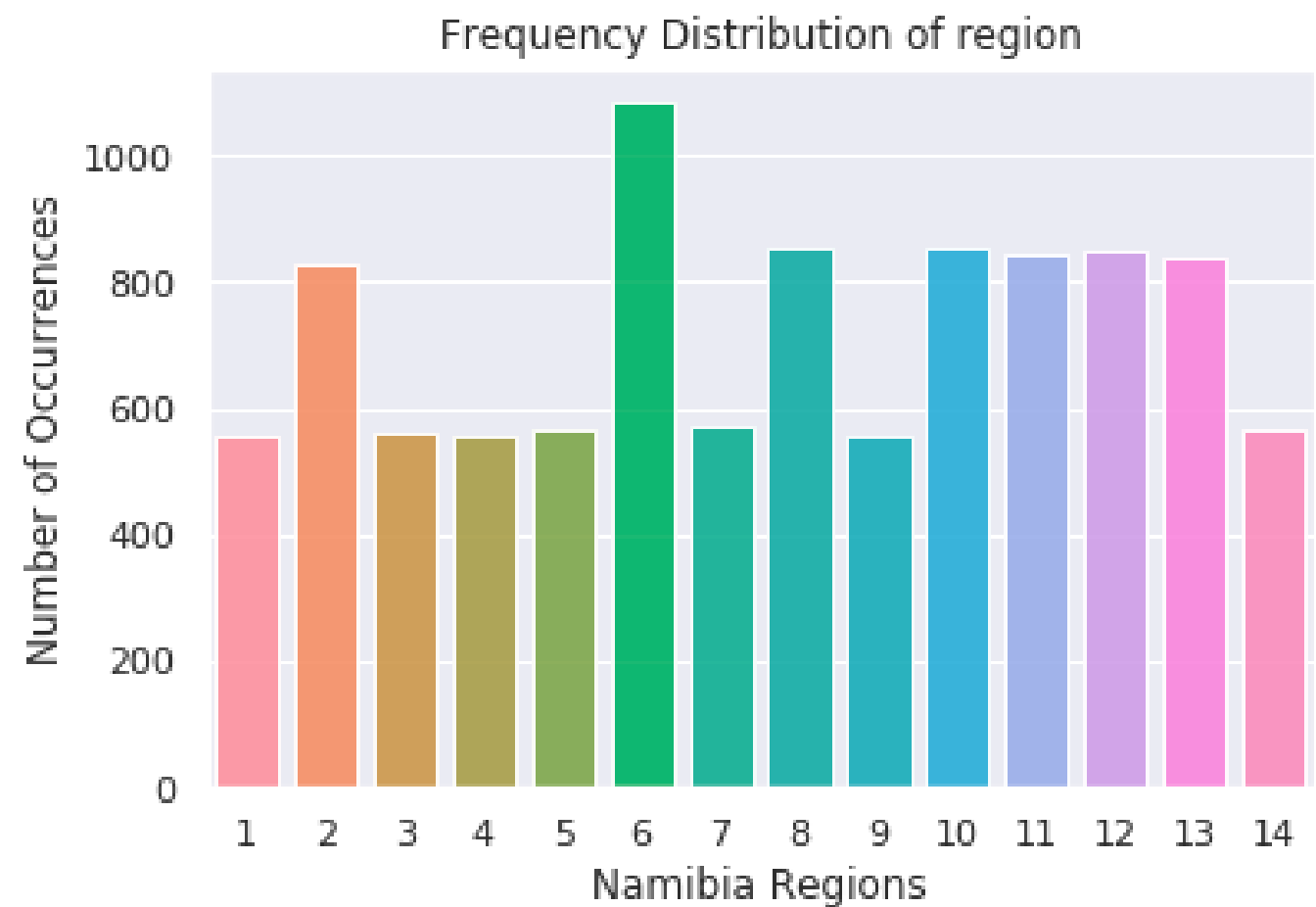
4:76-90

5:91-95

6:96-98

7:99-100

Region bar chart



- 1: !Karas
- 2: Erongo
- 3: Hardap
- 4: Kavango East
- 5: Kavango West
- 6: Khomas
- 7: Kunene
- 8: Ohangwena
- 9: Omaheke
- 10: Omusati
- 11: Oshana
- 12: Oshikoto
- 13: Otjozondjupa
- 14: Zambezi

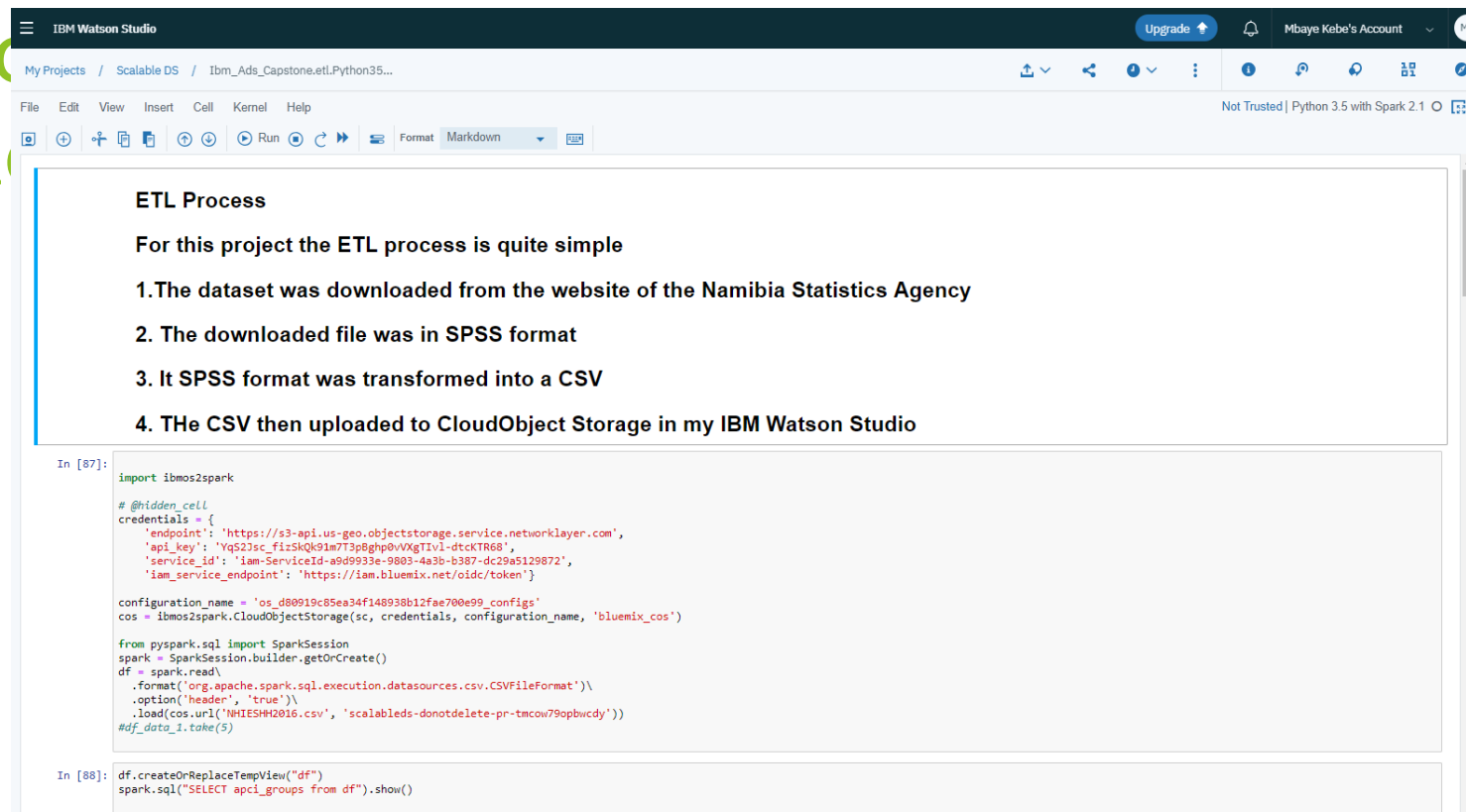
Architectural Decisions

- ▶ As already stated data set downloaded from NSA Data Portal
 - Data publicly available
 - Anonymized
 - Cleaned
- ▶ The process model was implemented on the IBM Watson and IBM Developer Skills Network Labs (CCLabs) on the Cloud.
 - Open source tools used (Jupyter, Python, pandas,matplotlib, scikit-learn, Keras, TensorFlow).
 - There is the possibility to utilize SystemML / Apache Spark to process data in multiple servers - was not used, though. Only part of ETL on Apache Spark.

Process Model Task 1 - Initial Data Exploration

- ▶ Very quickly in SPSS, Microsoft Excel before loading the data in IBM Cloud Object Storage as shown in the next slide.
- ▶ There no major data cleaning issues, since the data was already cleaned by the NSA.
- ▶ Hence we had good quality , anonymized data at our disposal to undertake the research.

Proc Load



The screenshot displays the IBM Watson Studio web interface. The top navigation bar includes the IBM Watson Studio logo, an 'Upgrade' button, a notification bell, and the user's account information 'Mbaye Kebe's Account'. Below the navigation bar, the breadcrumb trail shows 'My Projects / Scalable DS / Ibm_Ads_Capstone.etl.Python35...'. The main toolbar contains standard file editing and development icons, along with a 'Format' dropdown set to 'Markdown'. The notebook content is divided into two sections. The first section, titled 'ETL Process', contains a paragraph and a four-step numbered list. The second section contains two code cells. The first code cell, labeled 'In [87]:', contains Python code for connecting to IBM Cloud Object Storage and loading a CSV file into a Spark DataFrame. The second code cell, labeled 'In [88]:', contains code to create a temporary view and display the first five rows of the DataFrame.

ETL Process

For this project the ETL process is quite simple

1. The dataset was downloaded from the website of the Namibia Statistics Agency
2. The downloaded file was in SPSS format
3. It SPSS format was transformed into a CSV
4. The CSV then uploaded to CloudObject Storage in my IBM Watson Studio

```
In [87]:
import ibmos2spark

# @hidden_cell
credentials = {
    'endpoint': 'https://s3-api.us-geo.objectstorage.service.networklayer.com',
    'api_key': 'YqS2Jsc_fi:SkQk91m7T3pBgph0vVXgTivl-dtckTR68',
    'service_id': 'iam-ServiceId-a9d9933e-9803-4a3b-b387-dc29a5129872',
    'iam_service_endpoint': 'https://iam.bluemix.net/oidc/token'

configuration_name = 'os_d80919c85ea34f148938b12fae700e99_configs'
cos = ibmos2spark.CloudObjectStorage(sc, credentials, configuration_name, 'bluemix_cos')

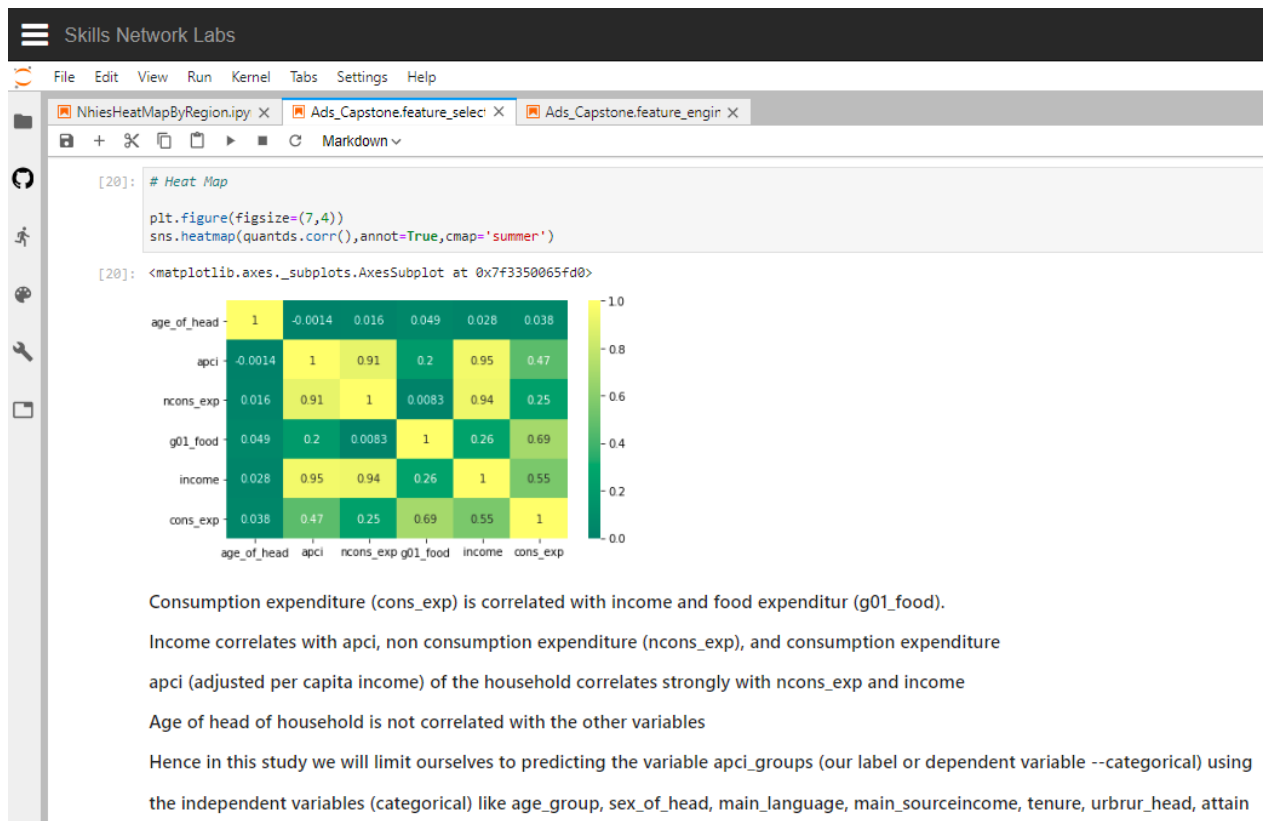
from pyspark.sql import SparkSession
spark = SparkSession.builder.getOrCreate()
df = spark.read\
    .format('org.apache.spark.sql.execution.datasources.csv.CSVFileFormat')\
    .option('header', 'true')\
    .load(cos.url('NMIESHH2016.csv', 'scalableds-donotdelete-pr-tmcow79opbwcdy'))
#df_data_1.take(5)

In [88]:
df.createOrReplaceTempView("df")
spark.sql("SELECT apci_groups from df").show()
```

Process Model Task 3 - Feature Creation/Engineering

- ▶ Two Python 3 Notebooks were created for the task in CC Labs on the IBM Cloud.
- ▶ In the first one, we looked at the quantitative variables
- ▶ In the second one we looked at the qualitative/categorical variables
- ▶ Lots of processing to get a Data Frame suitable for modeling.
 - ✓ Grouping
 - ✓ Filtering
 - ✓ Recoding columns
 - ✓ Encoding in forms suitable to modeling
 - ✓ All categorical data features "one-hot encoded" for the algorithms to work
- ▶ Multiple Correspondence Analysis, Correlation Analysis, Chi Square tests, and Cramer's V were used.
- ▶ That led to the selection of the final features and the label to be used for Model definition.

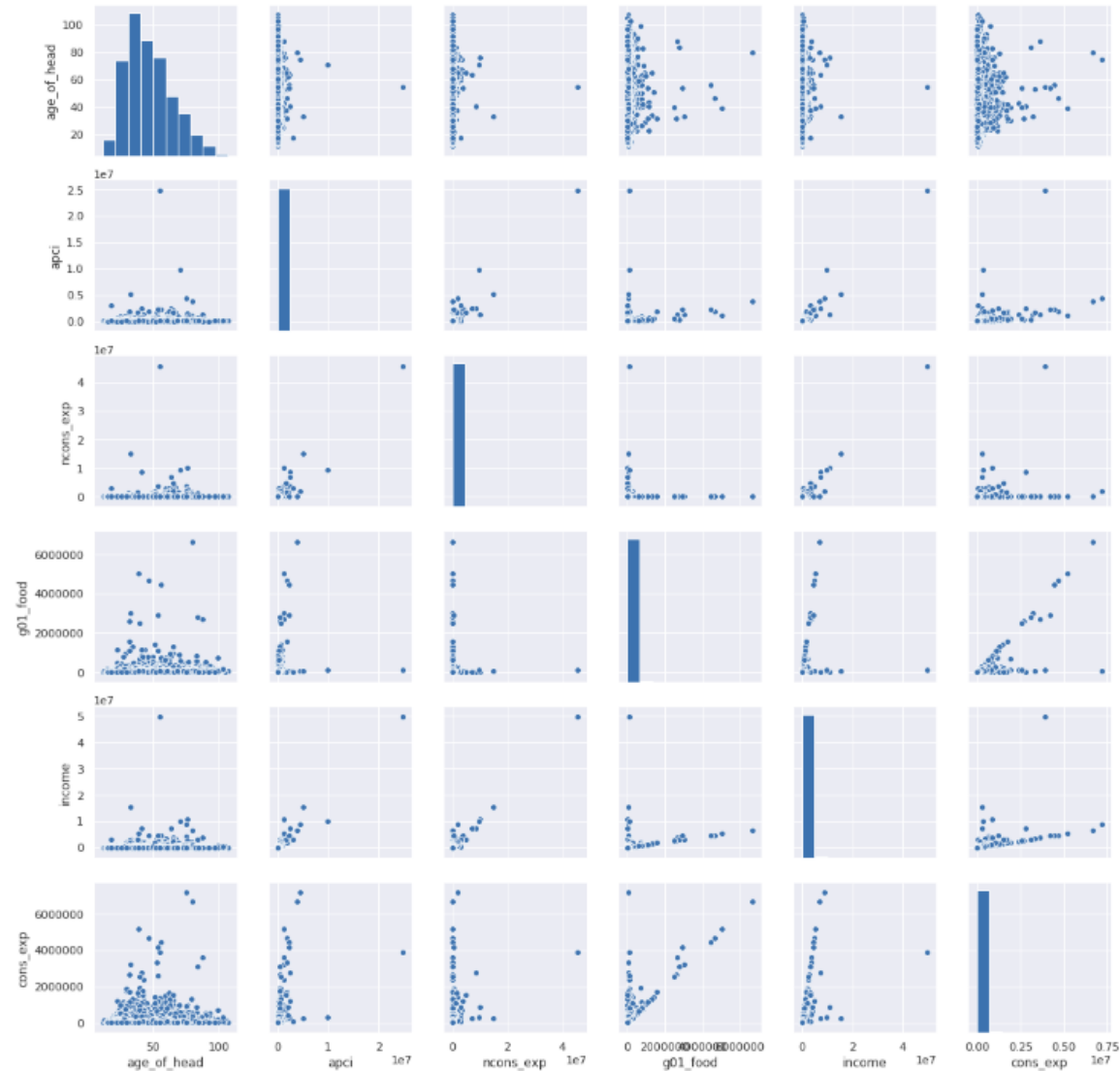
Correlations between quantitative variables



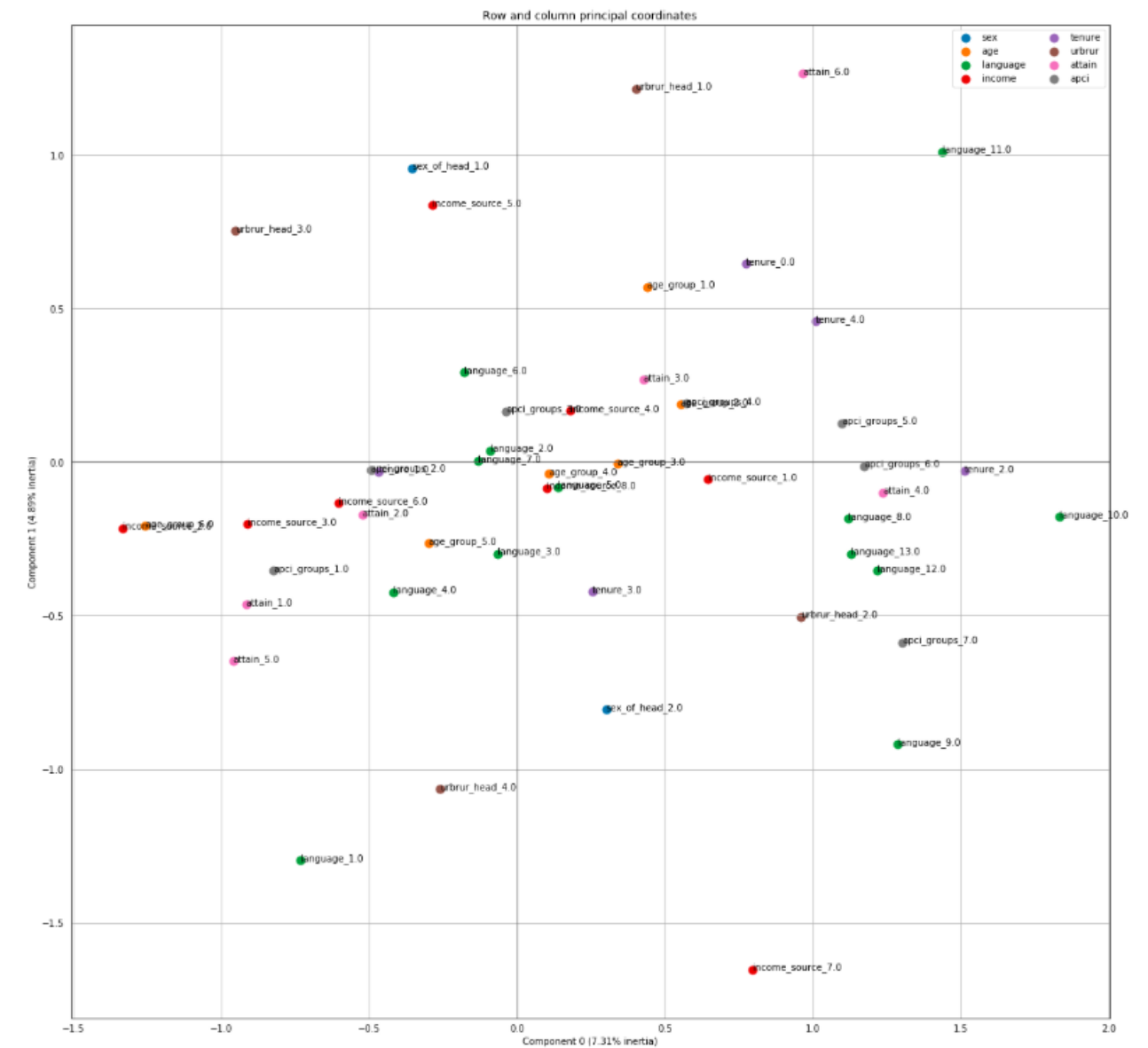
Pair plot between quantitative variables

```
[40]: sns.pairplot(quantds,height=2.5)
```

```
[40]: <seaborn.axisgrid.PairGrid at 0x7f3347645f28>
```

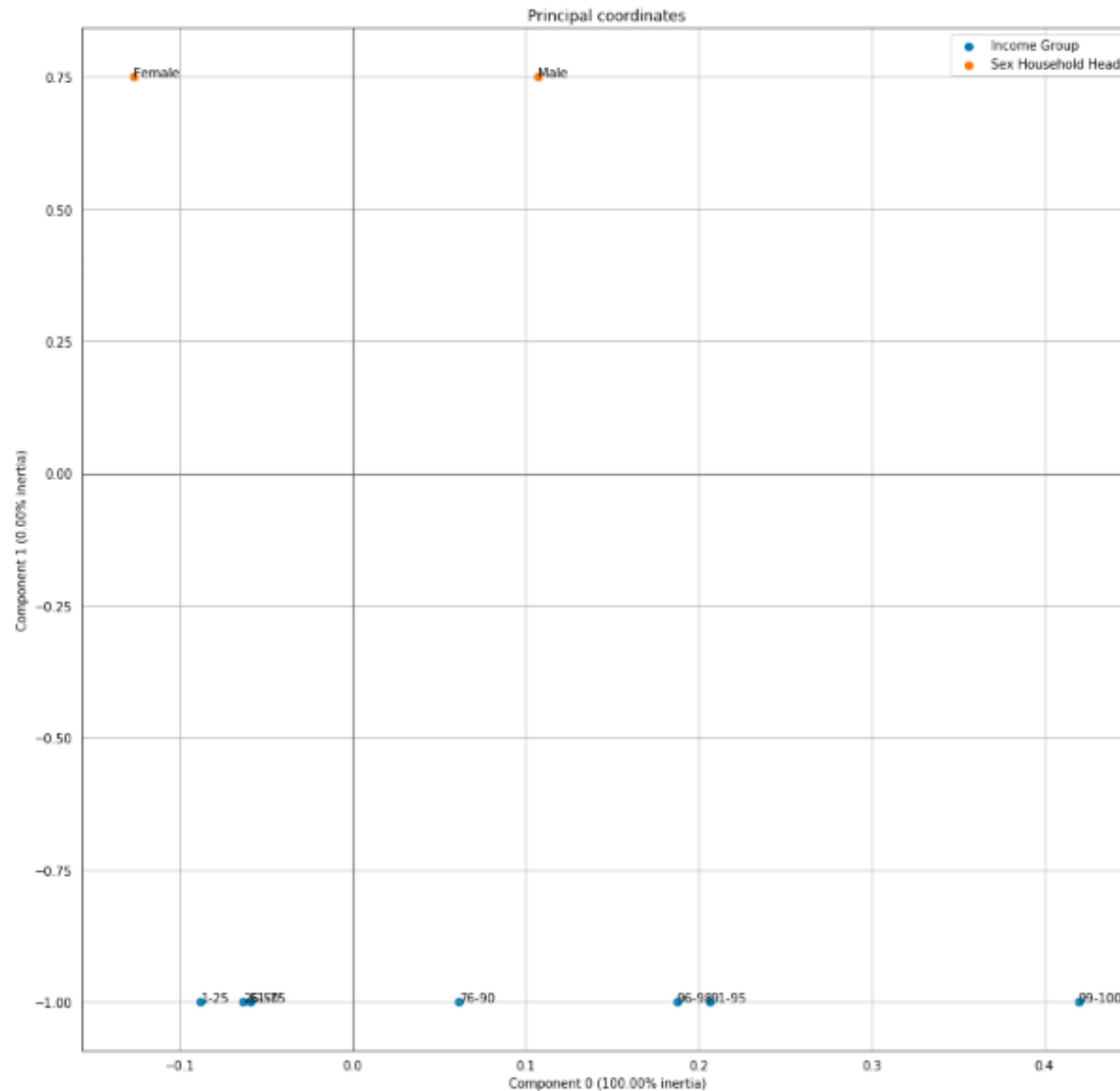


Association between categorical variables



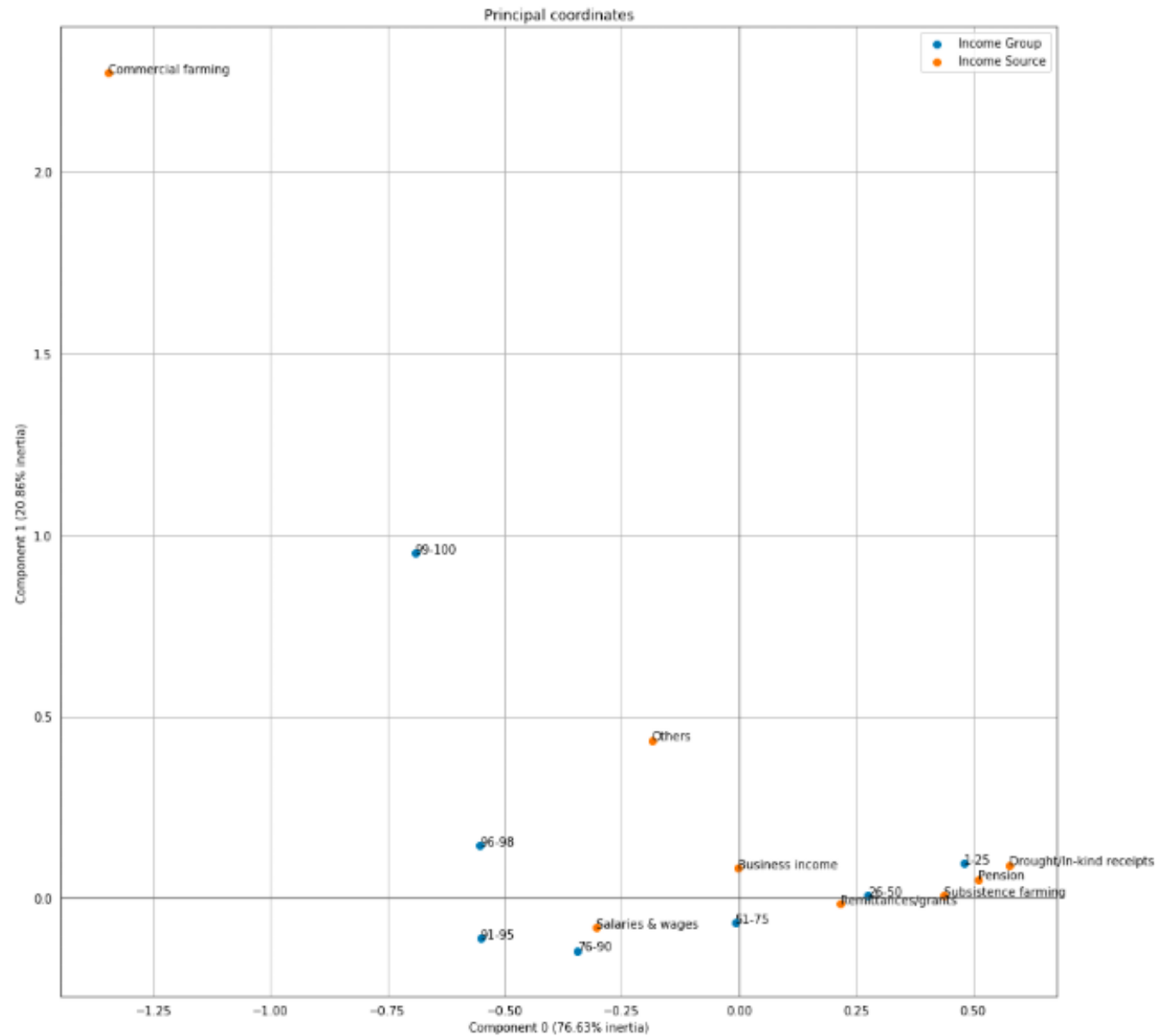
Association between categorical variables

income group sex of head of household

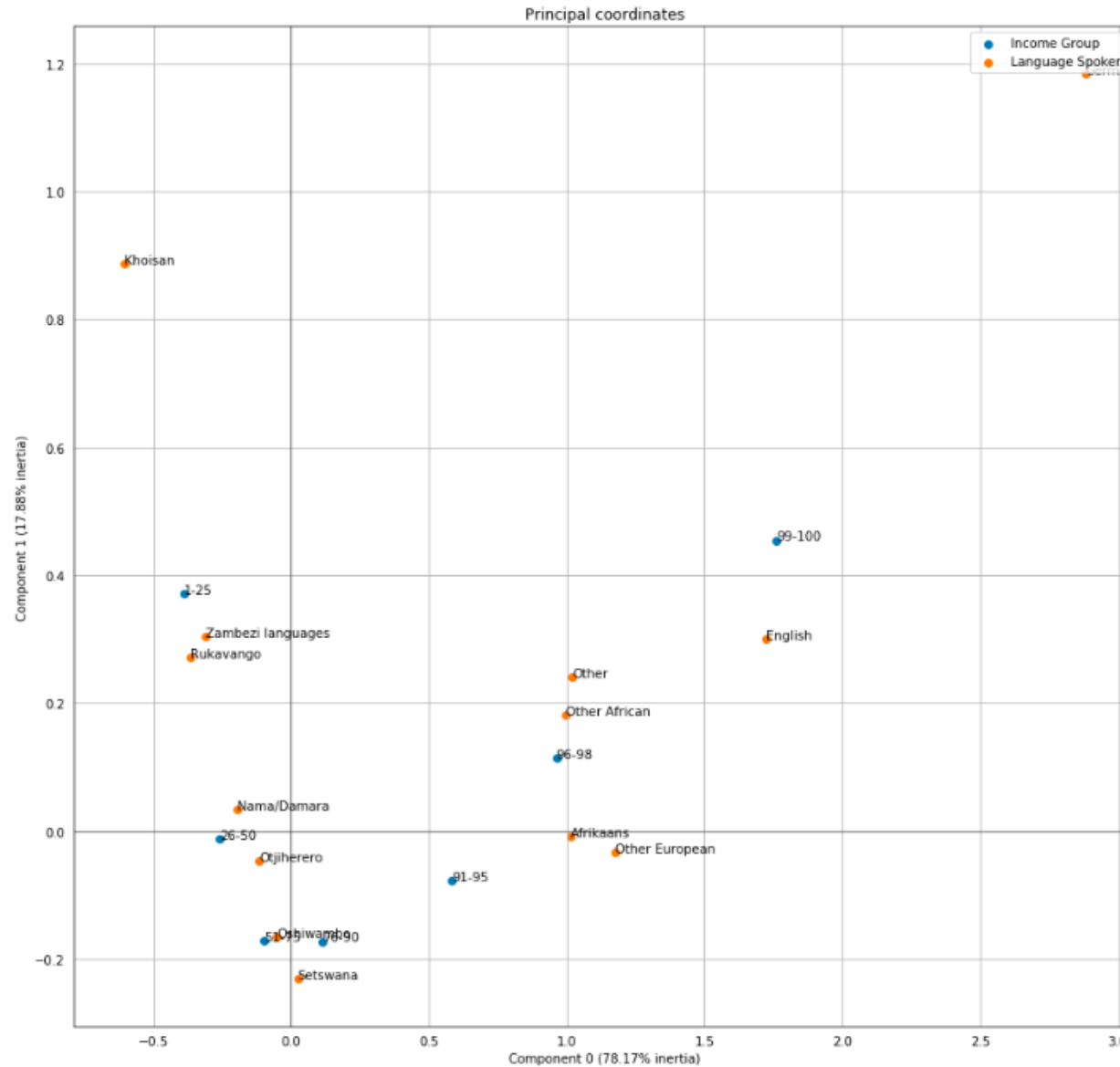


Clear separation between male headed and female headed households; male tend to lean towards the high income groups contrary to females.

Association between categorical variables income group and income source



Association between categorical variables income group and language spoken by head of household



Process Model Task 4 - Model Definition

- ▶ Based on the detective work performed during the previous tasks, the best potential features selected to predict our label/dependent - 'apci_groups' -- variable of interest were 'sex_of_head', 'income_source', 'language'
- ▶ Hence in formal terms our equation will be:
- ▶ $\text{apci_groups} = f(\text{sex_of_head}, \text{income_source}, \text{language})$
- ▶ In practice we will fit this model using Deep Learning technologies (Keras, TensorFlow, Theano) and non Deep Learning technologies such as kNN, GBTClassifier, Multinomial LogisticRegression, AdaBoostClassifier for comparison purposes.

The Model with Keras - Deep Learning

STEP 6 Import Modules for Model Development

```
[168]: # Importing the Keras libraries and packages
import keras
from keras.models import Sequential
from keras.layers import Dense
```

STEP 7 Initialize Sequential Model

```
[169]: #Initializing Neural Network
classifier = Sequential()
```

STEP 8 Adding the layers one by one

```
[170]: # first input layer
classifier.add(Dense(units = 5, kernel_initializer = 'uniform', activation = 'relu', input_dim = 36))
```

```
[171]: # hidden layer

classifier.add(Dense(units = 3, kernel_initializer = 'uniform', activation = 'relu'))
```

```
[172]: # output layer

classifier.add(Dense(units = 7, kernel_initializer = 'uniform', activation = 'softmax'))
```

The Model with non Deep Learning 1

STEP 5 kNN Model

```
[119]: from sklearn.model_selection import cross_val_score
```

```
[120]: from sklearn.neighbors import KNeighborsClassifier
```

```
[121]: # 10-fold cross validation with K=5 for KNN  
knn = KNeighborsClassifier(n_neighbors=15)  
scores = cross_val_score(knn, X,Y,cv=5,scoring='accuracy')  
scores.mean()
```

```
[121]: 0.9223005911058373
```

STEP 6 Logistic Regression Model

```
[129]: from sklearn.linear_model import LogisticRegression  
# 10-fold cross-validation with logistic regression  
logreg=LogisticRegression()  
# logreg.fit(X_train,y_train)  
scores = cross_val_score(logreg, X,Y,cv=10,scoring='accuracy')  
scores.mean()
```

The Model with non Deep Learning 2

STEP 7 GBClassifier , AdaBoostClassifier, RandomForestClassifier models

```
[130]: ## GBClassifier
```

```
[132]: from sklearn.ensemble import GradientBoostingClassifier
      clf = GradientBoostingClassifier(n_estimators=200, learning_rate=1.0, max_depth=1, random_state=0).fit(X_train, y_train)
      scores = cross_val_score(clf, X, Y, cv=10)
      scores.mean()
```

```
[132]: 0.9704662358173193
```

```
[133]: ## AdaBoostClassifier
      from sklearn.ensemble import AdaBoostClassifier
      clf = AdaBoostClassifier(n_estimators=100)
      scores = cross_val_score(clf, X, Y, cv=10)
      scores.mean()
```

```
[133]: 0.7006939933944802
```

```
[134]: from sklearn.ensemble import RandomForestClassifier

      rclf = RandomForestClassifier()
      scores = cross_val_score(rclf, X, Y, cv=10)
      scores.mean()
```

Process Model Task 5 - Model Training

- ▶ For Deep Learning, after splitting the dataset in a training and testing set, the model was trained using the training set.
- ▶ For the non Deep Learning, we used cross validation techniques for the training and testing.

Process Model Task 6 - Model Evaluation

	Deep Learning	Non Deep Learning				
	Sequential Model	kNN	Logistic Regression	GBClassifier	AdaBoostClassifier	RandomForestClassifier
Model Accuracy	0.980	0.922	0.708	0.970	0.701	0.972

Process Model Task 7 - Model Deployment

- ▶ As stated in the Architectural Decision Document, there will be a PDF document to show the results since there is no need for now to deploy the model in production. This can be easily done however in IBM Watson on the Cloud or using Node-RED if needed.

Conclusions

- ▶ The machine learning models have been used successfully to gain useful insights from a good quality dataset. We predicted with high accuracy the adjusted per capita income percentile group of a household based on the sex of the head of household, the income source of the household, and the language spoken by the head of household.
- ▶ The Deep Learning techniques performed better in terms of classification accuracy than their non deep learning counterparts.

Thank You!