Analysing Political Executives with AI

Shuai Zhao1, Shalendra Sharma and Jin Xu

Shenzhen University1, Lingnan University1,

[shuaizhao@ln.hk](mailto:shuaizhao@ln.hk), [shalendrasharma@ln.edu.hk](mailto:shalendrasharma@ln.edu.hk), jinxu@dgut.edu.cn

**Abstract**

This study uses advanced mathematical and statistical models to test a number of hypotheses regarding political executives. Specifically, it analyses the impact these powerful leaders have on economic growth by using leaders’ data from the Archigos database from 1835 to the end of 2015. The data is modelled by the AutoGluon, which was developed by Amazon through machine learning, and deep learning. Automated Machine Learning (AutoML) and AutoGluon automatically extract features from the data and then use multiple classifiers to train the data. Use a linear regression model and classification model to establish the relationship between leaders and economic growth (GDP per capita growth), and to clarify the relationship between their characteristics and economic growth from a machine learning perspective. Our work may show as a model or signal for collaboration between the fields of statistics and artificial intelligence (AI) that can light up the way for statisticians and economists.

(code can be found in github link: will show later after code review)

**1. Introduction**

Leadership is widely investigated in sociology, psychology, and organization theory, as well as in political science (Jones 1989). In political science, there are several clearly specified subfield, congressional leaders, executive leaders (presidents, governors, and mayors), party leaders, bureaucratic leaders, interest-group leaders, and leaders in various policy areas (Jones 1989). In this article, we mainly focus on executive leaders, especially presidents. Archigos database is famous among political scholars for research about political leaders. In conflict and war, relationship between regime type, the fate of leaders and war (Debs, A., & Goemans 2010), authoritarian leaders are inclined to fight wars longer than are democratic leaders; democratic leaders select wars that have a lower risk of defeat than their authoritarian counterparts (DeMesquita, B. B., & Siverson 1995). Similarly, how proliferation some 1,342 leaders in office from 1945 to 2000, addressed the problem of nuclear proliferation revealed similar trends (Fuhrmann, M., & Horowitz 2015). In terms of education, better educated leaders tend to focus more on increasing the quality of government (Besley, T., & Reynal-Querol 2011), and Western-educated leaders significantly improve a country’s democratization prospects or future (Gift, T., & Krcmaric 2017). With reference to policies, political executives' life experiences influence their policy choices when they acquire power (Horowitz, M. C., & Stam 2014), and international institutions influence on domestic politics (Baccini, L., & Urpelainen 2014). Leaders who are less likely to be replaced are less likely to reform institutions that may potentially constrain executive power (Besley, T., Persson, T., & Reynal‐Querol 2016). Leadership turnover and regime change are highly correlated with military force. (Wu, C. X., & Wolford 2018).

Some scholars use various mathematics and statistical methods to examine the relationship between leaders and economic development and growth. Using deaths of leaders while in office as a source of exogenous variation in leadership, which means leaders left power randomly due to either natural causes or an accident. To analyse whether these plausibly exogenous leadership transitions are associated with shifts in country growth rates. Knowing leaders matter for growth in general is very different from knowing which leaders matter for growth (Easterly, W., & Pennings 2018). Hence, some have developed a methodology to estimate the growth contribution of individual leaders and calculate its precision (Easterly, W., & Pennings 2020). Some have used the dataset including all national leaders in the post-World War II period, from 1945 to 2000, and growth data from Penn World Tables (Jones, B. F., & Olken 2005). Using the 57 leaders’ transitions, their results confirm that leaders matter. When a one standard deviation changes in leader quality, it will lead 1.5 percentage points change in growth changes (Jones, B. F., & Olken 2005). Their methodologies are, a standard Wald test and a nonparametric Rank test, which are quite complex. Timothy Besley, Jose G. Montalvo and Marta Reynal-Querol based on the results of Jones and Olken (2005), that leaders matter for growth. They used more than 1000 political leaders from Archigos dataset between 1875 and 2004 to clarify that educated leaders affect the rate of economic outcomes (Besley, T., Montalvo, J. G., & Reynal‐Querol 2011). Their data set has 215 leaders who exit office because of natural death or terminal illness, but only 158 leaders have the education information. They found that growth is more higher under political executives with higher education. They also use statistical model to estimate leaders’ quality to investigate whether leaders matter for economic performance and, in particular, whether more educated leaders generate higher growth.

The above discussion underscores that researchers not only use statistical methods to analyse leadership’s data, in recent decades machine learning method has been developed to handle large scale data. Machine learning (ML) solves the problems using computer algorithms to improve automatically through experience (Jordan, M. I., & Mitchell 2015). Linear regression is one of the most commonly used method in machine learning. It is the only one model, we can obtain analytical solutions through independent features and dependent feature. Based on two assumptions: (1), the relationship between independent features and dependent feature is linear; (2), the noise is well-behaved following a Gaussian distribution (Zhang, A., Lipton, Z. C., Li, M., & Smola 2021). Linear regression often provide an adequate and interpretable description of how the inputs affect the output and understanding linear models is essential for understanding nonlinear ones (Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman 2009). Not all prediction problems are regression problems. Classification can introduce nonlinear into our model. The goal of classification problems is to predict which of a set of classes the data belongs to. For an important class of procedures, these decision boundaries are linear, this is linear methods for classification (Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman 2009).

This research intends to use machine learning method to analyse their relationship, making the methodology more concise and easier to understand. Machine Learning methods are the hot topics in computer science disciplines, and have been adopted for a wide range of real-world applications, ranging from social networks, online image/video-sharing platforms, and e-commerce to education, healthcare, financial data analysts, etc. However, data scientists, use several components of machine learning methods, including data representation, hyperparameter, and model architecture, in order to achieve good performance. Automated Machine Learning (AutoML), which can automate the process of applying machine learning methods, has received much attention in both academy and industry recently (Li, Y., Wang, Z., Xie, Y., Ding, B., Zeng, K., & Zhang 2021). AutoML includes Tree-Based Pipeline Optimization Tool (TPOT) (Olson, Edu, and Moore 2016), H2O (LeDell, E., & Poirier 2020), AutoWEKA (Thornton et al, 2013), auto-sklearn (Feurer et al, 2015), AutoGluon (Erickson et al, 2020), and Google AutoML Tables. This paper will use AutoGluon to model the data about political leaders. Use AutoGluon as a baseline to get a significant relationship diagram to help statisticians and economists to model the data. ML as a supplement for statisticians and economists, as we know that statistical models are complex and hard to understand, ML can quickly give the scientists an intuition for the conjectures and the results, at last the results may guide the policies.

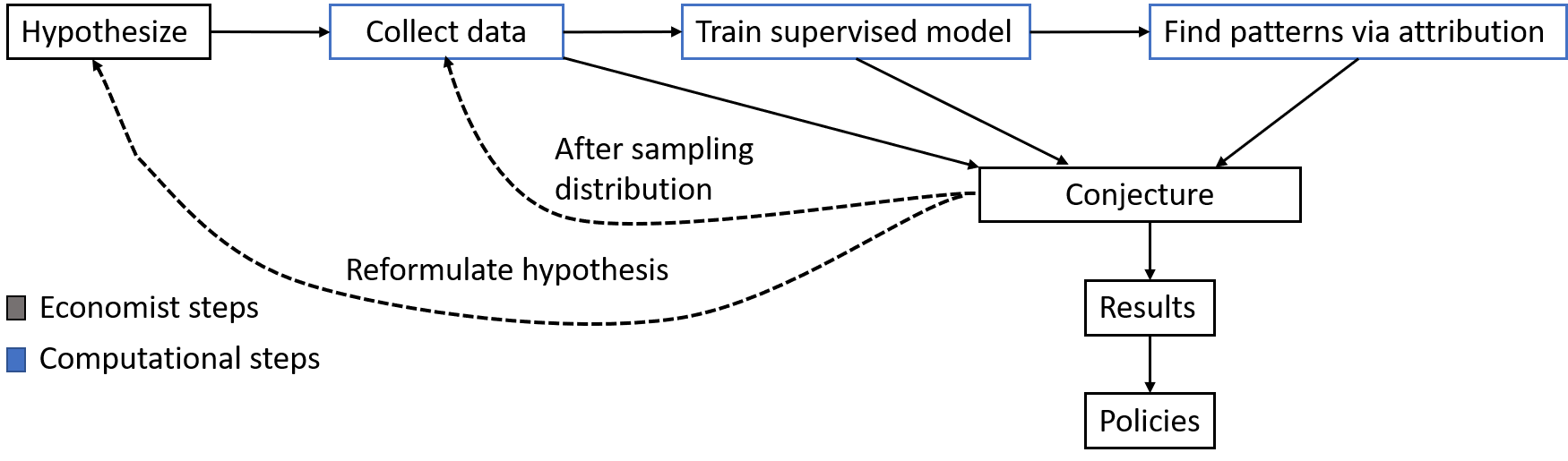


Figure 1. Flowchart of the framework

Figure 1., describes a general method for economist can use machine learning to guide their intuitions concerning complex large-scale data, verifying their hypotheses about the existence of relationships and helping them understand those relationships. We propose that this is a natural and empirically productive way that these well-understood techniques in statistics and machine learning can be used as part of an economist’s work. Generally, there are three steps for this process. First, data processing method and visualization can help us in the discovery of some plain relationship. Second, supervised model can be used in the ML process to verifying some hypotheses. At last, more advance model or unsupervised model will be used for proposing some unusual pattern by the ML itself.

**2. Data processing**

This paper uses data on political leaders from Archigos (Goemans, H. E., Gleditsch, K. S., & Chiozza 2009) dataset from 1835 to end 2015, and GDP data from Maddison Project Database 2020 (Bolt, J., & Van Zanden 2020). If countries have more than two leaders, the Archigos data use the effective power based on the characteristics of political system. And Archigos dataset has some refreshments since last version. This paper use Jupyter to analyse data and visualize results.

2.1 Import Data

Install the necessary Python packages, including Numpy, Pandas, Matplotlib, Seaborn, etc. The data format downloaded from the Archigos database is in ‘dta’ format, and Pandas can directly read that data format. After importing the data, we need to look at the overall summary of the data as a whole, as well as the header of the data, the data format of each column, etc. Table 1 shows the original data we imported from the database after we delete some useless features.

Table 1. List of original features

|  |  |
| --- | --- |
| Features | Explanation |
| obsid | country code with year |
| leadid | database provides each leader with a unique and stable leadid |
| ccode | country code |
| idacr | country code with 3 alphabets |
| leader | leader's name |
| startdate | acquire power year |
| eindate | acquire power year |
| enddate | leave power year |
| eoutdate | leave power year |
| entry | type (regular) |
| exit | type (regular/still in office) |
| exitcode | type (regular/still in office) |
| prevtimesinoffice | previous times in office |
| posttenurefate | post tenure fate |
| gender | male; female |
| yrborn | year was born |
| yrdied | year was died |
| borndate | date was born |
| ebirthdate | date was born |
| deathdate | date was died |
| edeathdate | date was died |
| dbpediauri | a stable DBPEDIA link to a wiki entry on the leader (Goemans, H. E., Gleditsch, K. S., & Chiozza 2016) |
| numentry | number of entry |
| numexit | number of exit |
| numexitcode | number of exit code |
| numposttenurefate | number of post tenure fate |
| fties | family ties |
| ftcur | an indicator of whether the tie is to a past leader (1) or a future leader (0) |

2.2 Data cleaning and preprocessing

Remove some data columns that have no practical use, and remove those columns where more than 30% of the data is null; modify the wrong data format. Check whether the data as a whole has outliers. The minimum value of yrborn and yrdied is negative, there is a problem with the data, and further data cleaning is required; new data columns are generated through the calculation between the data columns, such as the year leaders come into power and leave power yrbegin and yrend. Delete the data whose year is less than zero, including the leader's birth year, death year. Calculate the age of the leader when they enter office, year entering into office minus year of birth, yrbegin – yrborn.

Multi-table fusion: GDP data is in another file, we first combine the two data set through the same column, such as country name. We then calculate the growth\_rate = (end\_gdppc/begin\_gdppc-1)/tenure. After that, we delete the data with empty values in growth rate and leaders’ fields.

2.3 Data Standardization

Because machine learning model is very sensitive with numeric values, so we must deal with our data very carefully. For classification data, we give each group one number, and use rescaling (min-max normalization), to make sure all the data are between 0 and 1. For integer or float data with large difference, we make use of min-max normalization either. Normalization also known as standardization or feature scaling, is an essential step in data pre-processing in any machine learning application and model fitting, it will give ‘equal’ consideration for each feature. For classification, we try to avoid zero, because it will increase the bias and hardly to receive the exact features in the training process.

The distances between the maximum and minimum data of some features is quite different, it is not suitable to put it into the machine model for direct training. For gender, we name male as 1, and female as 0.5, to put them into the model. For the leaders who have family ties naming 1, without family ties naming 0.5, and with a new column named fties\_range. Table 2 is the detailed introduction of all the features in my model.

Table 2. List of features into AutoML model.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Features | Description | Types(original) | Explanation |
| 1 | leader | leader’s name | object | id |
| 2 | growth\_rate | (end\_gdppc/begin\_gdppc-1)/tenure | float64 | label1-linear regression |
| 3 | growth\_rate group | group growth\_rate into seven groups according to normal distribution | int64 | label2-normal classification |
| 4 | growth\_rate avggrp | divide growth\_rate into seven groups in descending order | int64 | label3-equivalent classification |
| 5 | obsid | country code with year | object | delete, it is the repetition of No.6 and 28 |
| 6 | leaderid | database provides each leader with a unique and stable leadid | object | delete, it is 1’s repeat |
| 7 | ccode | country code in number  ccode and idcar features from the Correlates of War project to identify countries  (Goemans, H. E., Gleditsch, K. S., & Chiozza 2016) | int16 | min-max normalization |
| 8 | idacr | country code with 3 alphabets  ccode and idacr features from the Correlates of War project to identify countries  (Goemans, H. E., Gleditsch, K. S., & Chiozza 2016) | object | delete, it is No.6’s repeat |
| 9 | startdate | date at which leader entered office in year month day format | object | delete, we extract the year as yrbegin |
| 10 | eindate | date at which leader entered office in year month day format | datetime64[ns] | delete, we extract the year as yrbegin |
| 11 | enddate | date at which leader left office in year month day format | object | delete, we extract the year as yrend |
| 12 | eoutdate | date at which leader left office in year month day format | datetime64[ns] | delete, we extract the year as yrend |
| 13 | entry | the manner in which a leader reaches power, factor with 4 levels (Regular 1; Irregular 2; Foreign Imposition 3; Unknown 4) | object | min-max normalization |
| 14 | exit | the manner with which a leader lost power, factor with 7 levels (Regular 1; Irregular 2; Foreign 3; Natural Death 4; Retired Due to III Death 5; Suicide 6; Unknown 7) | object | min-max normalization |
| 15 | exitcode | more detailed mode of exit, factor with 14 levels (Assassination by Unsupported Individual 1; Irregular, Other 2; Popular Protest, with Foreign Support 3; Popular Protest, without Foreign Support 4; Regular 5; Removed by Military, with Foreign Support 6; Removed by Military, without Foreign Support 7; Removed by Other Government Actors, with Foreign Support 8; Removed by Other Government Actors, without Foreign Support 9; Removed by Rebels, with Foreign Support 10; Removed by Rebels, without Foreign Support 11; Removed in Military Power Struggle Short of Coup 12; Removed through Threat of Foreign Force 13; Unknown 14) | object | min-max normalization |
| 16 | prevtimesinoffice | number of previous times in office, numeric (0, 1, 2, 3, 4, 5) | int8 | min-max normalization |
| 17 | posttenurefate | the fate of the leader in the period up to one year after the leader lost power, factor with 9 levels (Death 1; Exile 2; Imprisonment 3; Missing: Natural Death within Six Months of Losing Office 4; Missing: No Information Found 5; OK 6; Suicide 7) | object | min-max normalization |
| 18 | gender | gender, factor with 2 levels, M and F | object | male 1; female 0.5 |
| 19 | yrborn | leader’s birthyear, if known, numeric. | int16 | min-max normalization |
| 20 | yrdied | leader’s death year, if known, numeric. | int16 | min-max normalization |
| 21 | borndate | date of birth, if known, in year or year month day format | object | delete, we extract the year as yrborn |
| 22 | ebirthdate | date was born | datetime64[ns] | delete, we extract the year as yrborn |
| 23 | deathdate | format date of death, if known, in year or year month day | object | delete, we extract the year as yrdied |
| 24 | edeathdate | date was died | datetime64[ns] | delete, we extract the year as yrdied |
| 25 | dbpediauri | a stable DBPEDIA link to a wiki entry on the leader (Goemans, H. E., Gleditsch, K. S., & Chiozza 2016) | object | delete, we do not need the url information |
| 26 | numentry | number of entries (-666, 0, 1, 2) | int16 | min-max normalization |
| 27 | numexit | number of exits (-888, -666, 1, 2, 2.1, 2.2, 3, 4) | float32 | min-max normalization |
| 28 | numexitcode | number of exit code (-999, -888, 0, 1, 2,3, 4, 5, 6, 7, 8, 9, 11, 16, 111) | int16 | delete, do not really know about the meaning of the feature |
| 29 | numposttenurefate | number of post tenure fate (-999, -888, -777, -666, 0, 1, 2, 3, 3.1) | float32 | delete, do not really know about the meaning of the feature |
| 30 | fties | whether this leader was related through family ties to a previous leader, or is related to a future leader. The feature contains the relationship and the LEADID of the related leader | object | after naming with family ties 1, without family ties 0.5 in a new column fties\_range, delete, have more than 30% null values |
| 31 | ftcur | an indicator of whether the tie is to a past leader (1) or a future leader (0), blanks | object | delete, have more than 30% null values |
| 32 | yrbegin | year of tenure begin | int64 | min-max normalization |
| 33 | yrend | year of tenure end | int64 | min-max normalization |
| 34 | age | age reach power | int64 | min-max normalization |
| 35 | tenure | yrend-yrbegin | float64 | min-max normalization |
| 36 | Country | leaders’ country full name | object | delete, it is No.6’s repeat |
| 37 | polity datasets iv category | anocracy/democracy/autocracy | object | delete, have more than 30% null values |
| 38 | country\_x | leaders’ country | object | delete, it is No.32’s repeat |
| 39 | end\_gdppc | gdppc at the year leaders leave power | float64 | min-max normalization |
| 40 | pop\_x | people of population at the year leaders leave power | float64 | min-max normalization |
| 41 | country\_y | leaders’ country | object | delete, it is No.32’s repeat |
| 42 | begin\_gdppc | gdppc at the year leaders acquire power | float64 | min-max normalization |
| 43 | pop\_y | people of population at the year leaders acquire power | float64 | min-max normalization |
| 44 | fties\_range | have family ties or not | int64 | have family ties 1, otherwise 0.5 |

2.4 Data Correlation

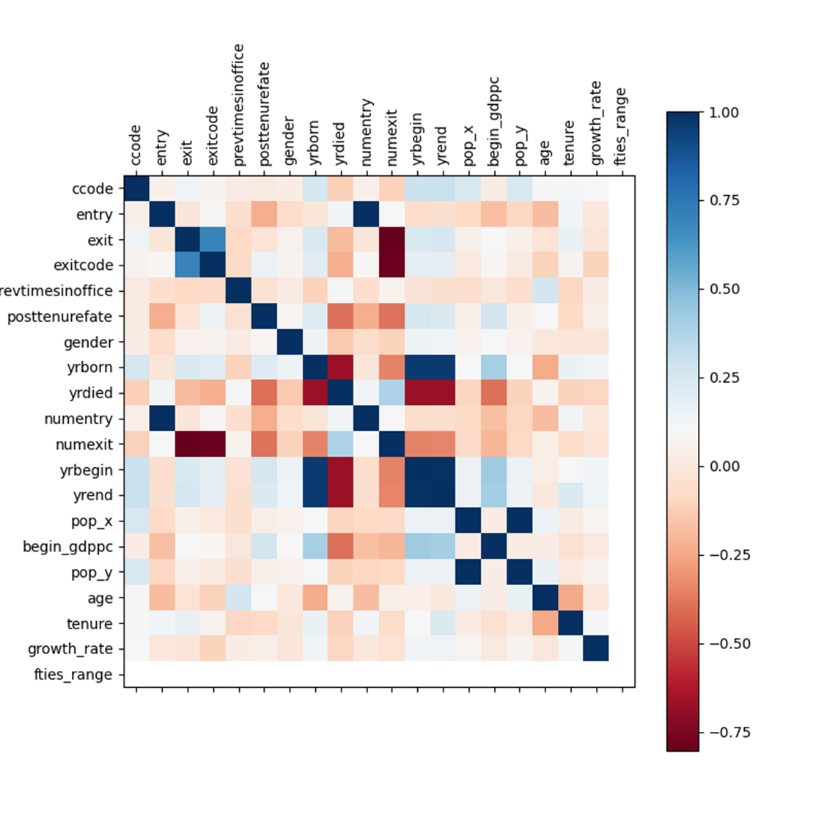


Figure 2. Correlation among features.

Figure 2., shows the correlation of all the features, which is calculated by pandas seaborn heatmap in Python. As you can see from the figure, dark blue represents the absolutely relevant with 1, and dark red means the very negative correlation with -0.6. Each feature is absolutely related to itself, such as the diagonal lines are dark blue; numentry has high positive relationship with entry; yrbegin, yrend, yrbegin, yrdied, they are strongly corelated with each other. So, we only choose meaningful ones as features in this research.

2.5 Data overview

Use seaborn to draw some overall pictures of our data. The following two charts show the leaders acquire power and leave power. There are four types that leaders acquire power: regular, irregular, foreign imposition and unknown. Regular has the largest proportion. Leaders leave power have eight different types: regular, irregular, natural death, still in office, retired due to ill health, foreign, suicide and unknown. Regular is also the most common in leaving office.

|  |  |
| --- | --- |
| Figure 3. Leaders acquire power | Figure 4. Leaders leave power |

Figure 5 shows the distribution of leaders’ age when they first come into power, leaders generally acquire power between 40 and 60 years old.

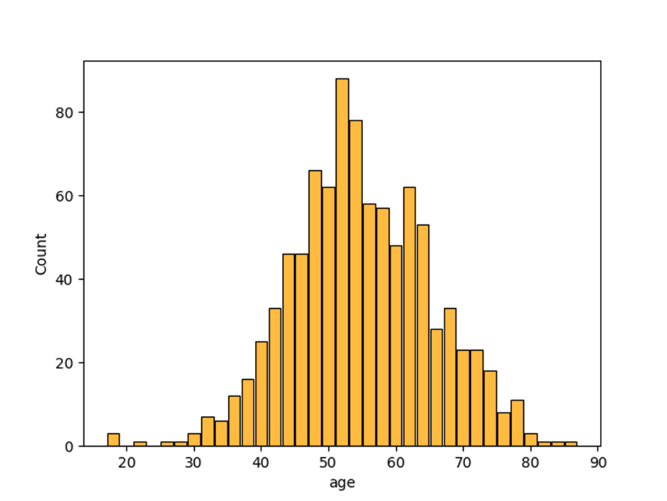


Figure 5. Distribution of leaders’ age acquire power

Calculate the tenure of the leader; the leaders’ tenure is the end time minus the start time, enddate – startdate; look at the overall distribution histogram of tenure. Most leaders have their tenures around 10 years, approximately 3 terms.

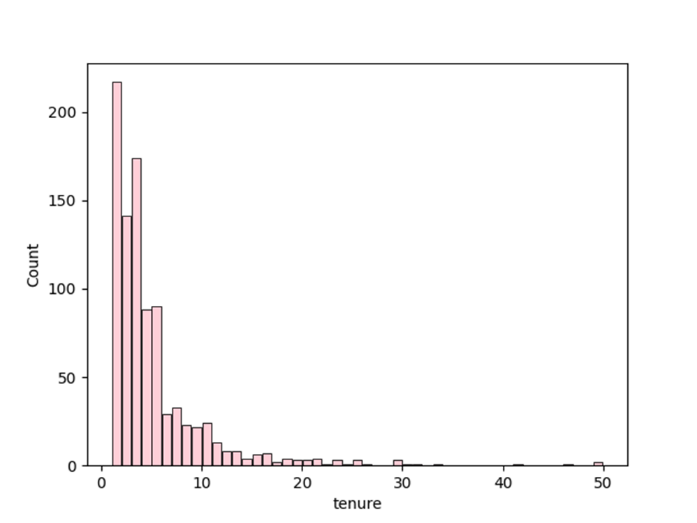


Figure 6. Distribution of tenure

We can see the distribution of GDP per capita growth rate from figure 7, which approximately follows a normal distribution.

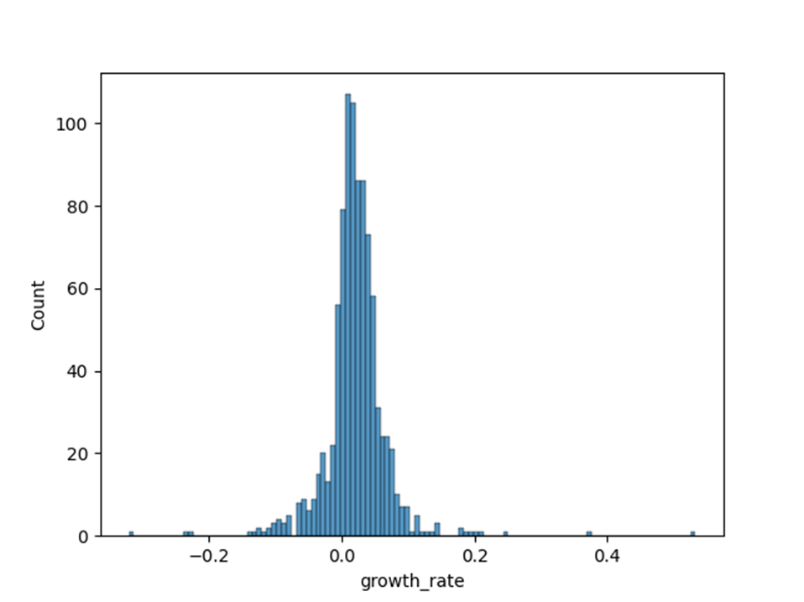


Figure 7. Distribution histogram of GDP per capita growth rate

**3. Modelling details**

The AutoGluon (Erickson, N., Mueller, J., Shirkov, A., Zhang, H., Larroy, P., Li, M., & Smola 2020) model is one of AutoML framework (automatically extract features from data and select appropriate models for training). Most AutoML frameworks are based on hyperparameter search technology, selecting a better model among dozens or hundreds of hyperparameter candidates, hoping to match the effect of manual feature tuning. AutoGluon relies on fusing multiple models without hyperparameter search, so that multiple different models can be trained at the same time.

Technique 1: Stacking

A number of different models are independently trained on the same data, which can be k-nearest neighbors (KNN), tree model, kernel method or complex neural network. After receiving the output of these models, they enter a linear model to obtain the final output, and make a weighted sum of these outputs, among which the weights are obtained through training. This is achieved through the model fusion method that using AutoGluon by default.

Technique 2: K-fold cross-bagging

Bagging refers to training multiple models of the same class, which may use different initial weights or different data blocks, and finally average the outputs of these models to reduce the variance of predictions. Derived from K-fold cross-validation.

Technique 3: Multi-layer stacking

Combine the output of each model with the data to do stacking again, and then train multiple models on this basis, and finally use a linear model to get the output. In order to avoid overfitting, the data in the later layers, multi-layer stacking needs to be used together with K-fold cross bagging.

Install the required packages, install our autogluon package. Use the package to train the model. We use the TabularDataset and TabularPredictor of autogluon tabular, and sklearn model\_selection to split the dataset. The id and label of our models are leader and growth\_rate and growth\_rate group (growth\_rate avggrp). We try to put all the features in the model, such as ‘leader’, ‘gender’, ‘age’, ‘tenure’, ‘fties\_range’, etc. Because of the limited data number, we put all data into training set. Figure 8 is AutoGluon’s multi-layer stacking strategy.

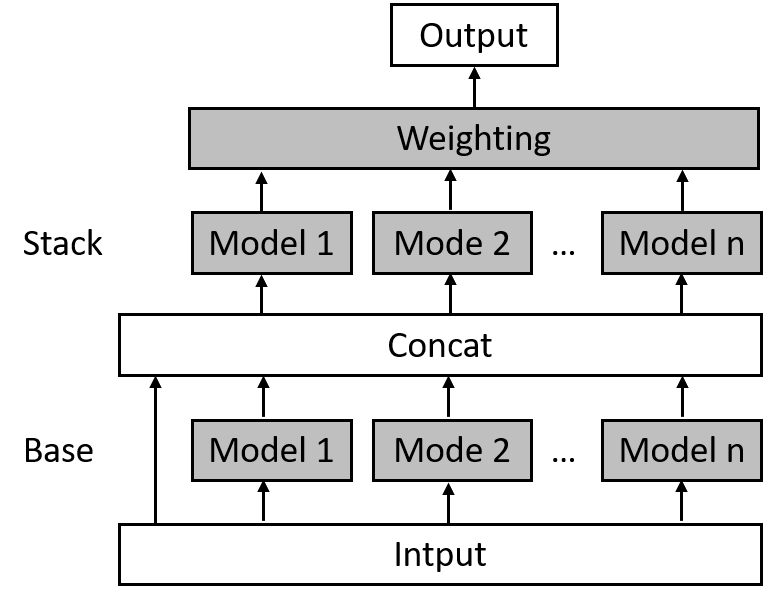


Figure 8. AutoGluon’s multi-layer stacking strategy.

Source: AutoGluon, 2020

**4. Results**

As shown in the flowchart of the framework, the ML models can be used in the economists’ pattern for finding process. Three kinds of models are used in this project. Linear regression model and classification model with normal distribution and equivalent distribution. The results are as follows.

**4.1 Linear regression**

Firstly, we try to put all the features into the model, but we got the results of the importance of each feature. If the feature has higher positive score, it means that feature is more important to the model’s performance; if a feature has a negative score, this means that the feature is likely harmful to the final model, and a model trained with the feature removed would be expected to achieve a better predictive performance (AutoGluon Tasks — AutoGluon Documentation 0.4.0 documentation n.d.).

Importance refers to techniques that assign a score to input features based on how useful they are at predicting a target feature. Stddev is standard deviation, a measure of the amount of variation. P\_value is the probability, if the null hypothesis is correct. N is the number of shuffles performed to estimate importance score (corresponds to sample-size used to determine confidence interval for true score). P99\_high and p99\_low is 99% confidence interval.

Table 3. Importance of features in linear regression.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **importance** | **stddev** | **p\_value** | **n** | **p99\_high** | **p99\_low** |
| ccode | 0.012403 | 0.000361 | 0.000141 | 3 | 0.014471 | 0.010336 |
| yrbegin | 0.010795 | 0.000661 | 0.000625 | 3 | 0.014585 | 0.007005 |
| yrend | 0.010314 | 0.000599 | 0.000562 | 3 | 0.013748 | 0.00688 |
| tenure | 0.009953 | 0.000414 | 0.000289 | 3 | 0.012328 | 0.007578 |
| yrborn | 0.009525 | 0.000424 | 0.00033 | 3 | 0.011955 | 0.007095 |
| yrdied | 0.008158 | 0.000329 | 0.000271 | 3 | 0.010044 | 0.006273 |
| age | 0.007565 | 0.000631 | 0.001157 | 3 | 0.011183 | 0.003947 |
| entry | 0.006895 | 0.00012 | 0.000051 | 3 | 0.007583 | 0.006206 |
| begin\_gdppc | 0.005877 | 0.000159 | 0.000122 | 3 | 0.006787 | 0.004967 |
| numentry | 0.005501 | 0.000121 | 0.00008 | 3 | 0.006192 | 0.00481 |
| posttenurefate | 0.004747 | 0.000052 | 0.00002 | 3 | 0.005045 | 0.004449 |
| exitcode | 0.004737 | 0.000828 | 0.005018 | 3 | 0.009483 | -0.000009 |
| pop\_x | 0.00377 | 0.000261 | 0.000797 | 3 | 0.005266 | 0.002274 |
| exit | 0.003671 | 0.000465 | 0.002651 | 3 | 0.006334 | 0.001007 |
| pop\_y | 0.003155 | 0.000203 | 0.00069 | 3 | 0.004319 | 0.001991 |
| prevtimesinoffice | 0.002821 | 0.000877 | 0.015366 | 3 | 0.007845 | -0.002204 |
| numexit | 0.001576 | 0.000661 | 0.026995 | 3 | 0.005366 | -0.002214 |
| gender | 0.000005 | 0.000003 | 0.048592 | 3 | 0.00002 | -0.000011 |

Model training process, the models with scores are chosen by the models are list as follows:

Table 4. Models used to train linear regression.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **model** | **score\_test** | **score\_val** | **pred\_time\_test** | **pred\_time\_val** | **fit\_time** | **pred\_time\_test\_marginal** | **pred\_time\_val\_marginal** | **fit\_time\_marginal** | **stack\_level** | **can\_infer** | **fit\_order** |
| 0 | KNeighborsDist | -0.01765 | -0.039402 | 0.03 | 0.012001 | 0.005998 | 0.03 | 0.012001 | 0.005998 | 1 | TRUE | 2 |
| 1 | XGBoost | -0.021361 | -0.044706 | 0.013 | 0.006 | 0.724512 | 0.013 | 0.006 | 0.724512 | 1 | TRUE | 9 |
| 2 | WeightedEnsemble\_L2 | -0.023972 | -0.036187 | 0.065001 | 0.031004 | 2.127913 | 0.007999 | 0.001005 | 0.306312 | 2 | TRUE | 12 |
| 3 | RandomForestMSE | -0.024779 | -0.040362 | 0.103 | 0.042002 | 0.573007 | 0.103 | 0.042002 | 0.573007 | 1 | TRUE | 5 |
| 4 | ExtraTreesMSE | -0.025145 | -0.039931 | 0.114534 | 0.039002 | 0.387283 | 0.114534 | 0.039002 | 0.387283 | 1 | TRUE | 7 |
| 5 | LightGBMLarge | -0.029761 | -0.037686 | 0.009001 | 0.006 | 0.717093 | 0.009001 | 0.006 | 0.717093 | 1 | TRUE | 11 |
| 6 | CatBoost | -0.034011 | -0.039171 | 0.00252 | 0.002998 | 1.030869 | 0.00252 | 0.002998 | 1.030869 | 1 | TRUE | 6 |
| 7 | LightGBM | -0.034961 | -0.037578 | 0.005 | 0.006 | 0.373999 | 0.005 | 0.006 | 0.373999 | 1 | TRUE | 4 |
| 8 | LightGBMXT | -0.039501 | -0.039008 | 0.004999 | 0.004 | 1.10436 | 0.004999 | 0.004 | 1.10436 | 1 | TRUE | 3 |
| 9 | KNeighborsUnif | -0.042279 | -0.040009 | 0.020979 | 0.019002 | 0.007999 | 0.020979 | 0.019002 | 0.007999 | 1 | TRUE | 1 |
| 10 | NeuralNetTorch | -0.044574 | -0.040961 | 0.024999 | 0.011997 | 4.23578 | 0.024999 | 0.011997 | 4.23578 | 1 | TRUE | 10 |
| 11 | NeuralNetFastAI | -0.045981 | -0.039404 | 0.032 | 0.016002 | 1.88863 | 0.032 | 0.016002 | 1.88863 | 1 | TRUE | 8 |

The leaderboard shows each model’s predictive performance on the test data (score\_test) and validation data (score\_val), as well as the time required to: produce predictions for the test data (pred\_time\_val), produce predictions on the validation data (pred\_time\_val), the fit time required to train the model end-to-end (fit\_time), inference time of the model for the data provided, minus the inference time for the model’s base models (pred\_time\_test\_marginal), inference time required to compute predictions on the validation data (pred\_time\_val\_marginal), it time required to train the model (Ignoring base models) (fit\_time\_marginal), stack level of the model (stack\_level), whether model is able to perform inference on new data or not (can\_infer), the order in which models were fit (fit\_order) (AutoGluon Predictors — AutoGluon Documentation 0.4.0 documentation n.d.).

NeuralNetFastAI, fastai is a deep learning library, which simplifies training fast and accurate neural nets using modern best practices, and provides practitioners with high-level components that can quickly and easily provide state-of-the-art results in standard deep learning domains (Welcome to fastai | fastai n.d.).

The evaluation of our linear regression, root mean squared error (MSE) on test data is -0.05404. The MSE is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are the less the better, that means most of the data are on the linear line. In AutoGluon program, scores are always higher is better. This score can be multiplied by -1 to get the metric value.

**4.2 Normal distribution classification**

We group the growth rate into seven different groups, according to normal distribution. The reason, why normal distribution classification method is used, is that the similarity of growth rate’s shape to normal distribution is high. We hope the normal distribution method can show a good prediction result. For the normal distribution, the values less than one standard deviation away from the mean account for 68.27% of the set; while two standard deviations from the mean account for 95.45%; and three standard deviations account for 99.73%. So, we divide our growth\_rate into the following seven groups: (-∞, -mean-3\*std), [-mean-3\*std, -mean-2\*std), [-mean-2\*std, -mean-std), [-mean-std, mean+std), [mean+std, mean+2\*std), [mean+2\*std, mean+3\*std), [mean+3\*std, +∞).

Table 5. Importance of features in normal distribution classification.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **importance** | **stddev** | **p\_value** | **n** | **p99\_high** | **p99\_low** |
| tenure | 0.03716 | 0.003548 | 0.001513 | 3 | 0.057491 | 0.016829 |
| ccode | 0.03532 | 0.006145 | 0.00497 | 3 | 0.070534 | 0.000106 |
| pop\_x | 0.032745 | 0.002298 | 0.000819 | 3 | 0.04591 | 0.019579 |
| pop\_y | 0.031641 | 0.000637 | 0.000068 | 3 | 0.035292 | 0.027989 |
| begin\_gdppc | 0.030905 | 0.001912 | 0.000637 | 3 | 0.04186 | 0.019951 |
| yrborn | 0.027226 | 0.003548 | 0.002807 | 3 | 0.047557 | 0.006895 |
| age | 0.023179 | 0.00398 | 0.004842 | 3 | 0.045983 | 0.000375 |
| yrdied | 0.022811 | 0.002778 | 0.002453 | 3 | 0.038728 | 0.006894 |
| yrbegin | 0.020603 | 0.002549 | 0.002532 | 3 | 0.035209 | 0.005997 |
| exit | 0.015453 | 0.00292 | 0.005848 | 3 | 0.032186 | -0.001281 |
| entry | 0.015085 | 0.000637 | 0.000297 | 3 | 0.018736 | 0.011433 |
| yrend | 0.014717 | 0.004461 | 0.014643 | 3 | 0.040277 | -0.010844 |
| prevtimesinoffice | 0.005151 | 0.000637 | 0.002532 | 3 | 0.008802 | 0.001499 |
| numexit | 0.005151 | 0.000637 | 0.002532 | 3 | 0.008802 | 0.001499 |
| posttenurefate | 0.004783 | 0.002778 | 0.048219 | 3 | 0.0207 | -0.011134 |
| exitcode | 0.00184 | 0.002778 | 0.18503 | 3 | 0.017756 | -0.014077 |
| gender | 0 | 0 | 0.5 | 3 | 0 | 0 |
| numentry | 0 | 0 | 0.5 | 3 | 0 | 0 |

Table 6. Models used to train normal distribution classification.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **model** | **score\_test** | **score\_val** | **pred\_time\_test** | **pred\_time\_val** | **fit\_time** | **pred\_time\_test\_marginal** | **pred\_time\_val\_marginal** | **fit\_time\_marginal** | **stack\_level** | **can\_infer** | **fit\_order** |
| 0 | LightGBM | 0.964208 | 0.906593 | 0.048999 | 0.006999 | 1.016001 | 0.048999 | 0.006999 | 1.016001 | 1 | TRUE | 5 |
| 1 | KNeighborsDist | 0.963124 | 0.901099 | 0.023999 | 0.014004 | 0.005998 | 0.023999 | 0.014004 | 0.005998 | 1 | TRUE | 2 |
| 2 | ExtraTreesGini | 0.963124 | 0.901099 | 0.091 | 0.062997 | 0.448002 | 0.091 | 0.062997 | 0.448002 | 1 | TRUE | 9 |
| 3 | RandomForestEntr | 0.963124 | 0.901099 | 0.096 | 0.054999 | 0.464 | 0.096 | 0.054999 | 0.464 | 1 | TRUE | 7 |
| 4 | ExtraTreesEntr | 0.963124 | 0.901099 | 0.098999 | 0.052001 | 0.447002 | 0.098999 | 0.052001 | 0.447002 | 1 | TRUE | 10 |
| 5 | LightGBMXT | 0.962039 | 0.912088 | 0.025 | 0.007002 | 0.903996 | 0.025 | 0.007002 | 0.903996 | 1 | TRUE | 4 |
| 6 | WeightedEnsemble\_L2 | 0.962039 | 0.912088 | 0.032999 | 0.007002 | 1.185193 | 0.007999 | 0 | 0.281196 | 2 | TRUE | 14 |
| 7 | RandomForestGini | 0.962039 | 0.895604 | 0.093 | 0.052003 | 0.459 | 0.093 | 0.052003 | 0.459 | 1 | TRUE | 6 |
| 8 | LightGBMLarge | 0.957701 | 0.906593 | 0.012 | 0.004998 | 1.625069 | 0.012 | 0.004998 | 1.625069 | 1 | TRUE | 13 |
| 9 | NeuralNetFastAI | 0.903471 | 0.906593 | 0.032003 | 0.013849 | 1.043209 | 0.032003 | 0.013849 | 1.043209 | 1 | TRUE | 3 |
| 10 | XGBoost | 0.896963 | 0.895604 | 0.008001 | 0.006999 | 0.811001 | 0.008001 | 0.006999 | 0.811001 | 1 | TRUE | 11 |
| 11 | KNeighborsUnif | 0.892625 | 0.906593 | 0.024 | 0.014 | 0.008998 | 0.024 | 0.014 | 0.008998 | 1 | TRUE | 1 |
| 12 | CatBoost | 0.886117 | 0.901099 | 0.004 | 0.003 | 1.302016 | 0.004 | 0.003 | 1.302016 | 1 | TRUE | 8 |
| 13 | NeuralNetTorch | 0.886117 | 0.901099 | 0.022997 | 0.019001 | 1.569999 | 0.022997 | 0.019001 | 1.569999 | 1 | TRUE | 12 |

CatBoost is an algorithm for gradient boosting on decision trees, and it is widely used for search, recommendation systems, personal assistant, self-driving cars, weather prediction and many other tasks at Yandex and in other companies, including CERN, Cloudflare, Careem taxi (CatBoost - open-source gradient boosting library n.d.).

The evaluation of our normal distribution classification, accuracy on test data is 0.87900.

**4.3 Equivalent classification**

We group the growth rate into seven different groups, according to descending order, and divide them into seven groups, each group with the same number of leaders, the rest we put into the last group. We give them the following numbers from 1 to 7 for each group.

Table 7. Importance of features in average classification.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **importance** | **stddev** | **p\_value** | **n** | **p99\_high** | **p99\_low** |
| begin\_gdppc | 0.053846 | 0.00479 | 0.001314 | 3 | 0.081293 | 0.026399 |
| ccode | 0.03663 | 0.005193 | 0.003317 | 3 | 0.066388 | 0.006872 |
| yrborn | 0.028205 | 0.006052 | 0.007502 | 3 | 0.062885 | -0.006475 |
| pop\_x | 0.025641 | 0.00416 | 0.004331 | 3 | 0.04948 | 0.001802 |
| tenure | 0.024542 | 0.00416 | 0.004722 | 3 | 0.048382 | 0.000703 |
| yrdied | 0.022711 | 0.003859 | 0.004744 | 3 | 0.044824 | 0.000597 |
| yrbegin | 0.021612 | 0.002288 | 0.001857 | 3 | 0.03472 | 0.008504 |
| yrend | 0.020513 | 0.003532 | 0.004871 | 3 | 0.040754 | 0.000271 |
| age | 0.016484 | 0.002198 | 0.002937 | 3 | 0.029077 | 0.00389 |
| pop\_y | 0.013187 | 0.003297 | 0.010102 | 3 | 0.032077 | -0.005704 |
| posttenurefate | 0.004029 | 0.000634 | 0.004082 | 3 | 0.007665 | 0.000394 |
| numexit | 0.002564 | 0.000634 | 0.009902 | 3 | 0.0062 | -0.001071 |
| exitcode | 0.002198 | 0.001099 | 0.03709 | 3 | 0.008495 | -0.004099 |
| exit | 0.001465 | 0.001269 | 0.091752 | 3 | 0.008736 | -0.005806 |
| entry | 0.000733 | 0.000634 | 0.091752 | 3 | 0.004368 | -0.002903 |
| numentry | 0.000733 | 0.000634 | 0.091752 | 3 | 0.004368 | -0.002903 |
| prevtimesinoffice | 0.000366 | 0.000634 | 0.211325 | 3 | 0.004002 | -0.003269 |
| gender | 0 | 0 | 0.5 | 3 | 0 | 0 |

Table 8. Models used to train average classification.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **model** | **score\_test** | **score\_val** | **pred\_time\_test** | **pred\_time\_val** | **fit\_time** | **pred\_time\_test\_marginal** | **pred\_time\_val\_marginal** | **fit\_time\_marginal** | **stack\_level** | **can\_infer** | **fit\_order** |
| 0 | RandomForestGini | 0.947939 | 0.802198 | 0.090002 | 0.056 | 0.506543 | 0.090002 | 0.056 | 0.506543 | 1 | TRUE | 6 |
| 1 | RandomForestEntr | 0.947939 | 0.802198 | 0.097001 | 0.052515 | 0.476093 | 0.097001 | 0.052515 | 0.476093 | 1 | TRUE | 7 |
| 2 | WeightedEnsemble\_L2 | 0.947939 | 0.802198 | 0.097999 | 0.056 | 0.790806 | 0.007997 | 0 | 0.284263 | 2 | TRUE | 14 |
| 3 | ExtraTreesEntr | 0.94577 | 0.791209 | 0.096998 | 0.060001 | 0.437003 | 0.096998 | 0.060001 | 0.437003 | 1 | TRUE | 10 |
| 4 | ExtraTreesGini | 0.944685 | 0.785714 | 0.091 | 0.051999 | 0.449001 | 0.091 | 0.051999 | 0.449001 | 1 | TRUE | 9 |
| 5 | LightGBMLarge | 0.938178 | 0.796703 | 0.013001 | 0.003 | 1.839508 | 0.013001 | 0.003 | 1.839508 | 1 | TRUE | 13 |
| 6 | KNeighborsDist | 0.937093 | 0.747253 | 0.026001 | 0.014492 | 0.00703 | 0.026001 | 0.014492 | 0.00703 | 1 | TRUE | 2 |
| 7 | XGBoost | 0.830803 | 0.791209 | 0.009 | 0.007 | 0.950999 | 0.009 | 0.007 | 0.950999 | 1 | TRUE | 11 |
| 8 | LightGBM | 0.81128 | 0.796703 | 0.004998 | 0.003001 | 0.673513 | 0.004998 | 0.003001 | 0.673513 | 1 | TRUE | 5 |
| 9 | CatBoost | 0.806941 | 0.802198 | 0.003995 | 0.000998 | 1.720608 | 0.003995 | 0.000998 | 1.720608 | 1 | TRUE | 8 |
| 10 | LightGBMXT | 0.804772 | 0.796703 | 0.008998 | 0.004 | 0.64202 | 0.008998 | 0.004 | 0.64202 | 1 | TRUE | 4 |
| 11 | NeuralNetFastAI | 0.793926 | 0.791209 | 0.030001 | 0.014 | 0.854465 | 0.030001 | 0.014 | 0.854465 | 1 | TRUE | 3 |
| 12 | KNeighborsUnif | 0.781996 | 0.741758 | 0.026002 | 0.015544 | 0.008001 | 0.026002 | 0.015544 | 0.008001 | 1 | TRUE | 1 |
| 13 | NeuralNetTorch | 0.773319 | 0.785714 | 0.026 | 0.019002 | 1.59351 | 0.026 | 0.019002 | 1.59351 | 1 | TRUE | 12 |

The evaluation of our average classification, accuracy on test data is 0.96085.

**5. Discussion**

The interpretation of the result that the root mean squared error of linear regression is negative: the root mean squared error of linear regression and the importance of each feature in the model is not that good. For linear regression y = <w\*x> + b, our label y (growth\_rate) is a little bit small ranged between -0.4 to +0.5, so w is much smaller than our y.

Use a linear regression model of AutoGluon to establish the relationship between leaders and economic growth (GDP per capita growth), and to clarify the relationship between their characteristics and economic growth by machine learning perspective. As a comparison, we also use classification introducing nonlinearity, to examine the relationship between leaders and the growth rate.

Accuracy is one metric for evaluating classification models. Accuracy is the ratio of the number of correct predictions divided by total number of predictions. Normal distribution classification has large differences in each group, but equivalent classification has small differences in each group, so the evaluation: accuracy on test data of equivalent classification is 0.96, normal distribution classification is 0.88.

Figure 9. Features importance in three different models

In the linear regression, the six more important influencing factors model are: yrend, exitcode, begin\_gdppc, yrborn, yrbegin, ccode. In normal distribution classification, the six dominant influencing factors are: yrend, yrbegin, yrborn, yrdied, exit, tenure. In equivalent classification, there are ccode, pop\_y, pop\_x, yrbegin, numexit, begin\_gdppc. And ccode plays a significant role in the model, with importance 0.822064. Because ccode is the specific country of leaders, that means country plays an important role in our normal distribution classification model. There are some features, which have no influence on our model. For normal distribution classification, there are numentry, begin\_gdppc, numexit, fties\_range, gender, posttenurefate, prevtimesinoffice, exitcode, entry. For equivalent classification, there are age, numentry, gender, prevtimesinoffice, exit, entry, fties\_range.

In the linear regression model, mathematicians say, we assume that the target (GDP growth rate) can be expressed as a weighted sum of the features (details in Table 2). This means the linear regression model can only fit the linear relationship. Furthermore, in the classification models, with the help of which active layer (with ReLU (Agarap 2018) or sigmoid (Han, J., & Moraga 1995) activation function), Softmax layer (Tüske, Z., Tahir, M. A., Schlüter, R., & Ney n.d.) and end with cross-entropy loss function (Ho, Y., & Wookey 2019), have the ability to fit non-linear models. That is the main reason why the classification model shows better features importance than the linear regression model. Because of enough negative samples, the equivalent classification shows better score sources than the norm classification with limited total samples numbers. With the total samples number increasing from thousands to millions, equivalent classification can show a better result than the norm classification method (Zhang, A., Lipton, Z. C., Li, M., & Smola 2021).

Researchers from economics examine this topic in different realms. Especially developing countries show sharper changes in growth patterns, and leaders have a large causative influence on economic performances (Jones, B. F., & Olken 2005), but which growth-related policies are the causing factors are still unknown due to limited data available (Besley, T., Montalvo, J. G., & Reynal‐Querol 2011). Less than 50 out of around 750 leaders with a tenure of at least 3 years with a tenure of more than 3 years confirmed significant leader effects (Easterly, W., & Pennings 2018), later research found that this figure increased to around 7%, and “benevolent autocrats” and “bad emperors” show significant leader effects (Easterly, W., & Pennings 2020).

Our paper explores leader effects on economic growth from a machine learning perspective. We use AutoGluon to receive the importance of each feature, and AutoGluon can give us the models used in the AutoGluon training process. We can compare the results of different training processes. This solution process would tell ordering importance of the leadership dataset. This case demonstrates how non-trivial insights and pattern of large statistical objects can be obtained from trained models.

**6. Conclusion**

This research intends to give statistician and economists an alternative method to think of the machine learning method used to explore political matters. We use the AutoGluon program to build models for research the relationship between political leaders and economic growth, and receive the following three conclusions.

1). Machine learning model is very sensitive to numeric values, so we must deal with our data very carefully. Data cleaning and data pre-processing account for 80% time of data scientists. Data preparation is the first and most important part of our analysis. For numeric data, if the data have a large difference in the distribution, we should standardize the data using scaling (min-max normalization). For classification data, we give each group one number to represent, and then use rescaling to make all the data are between 0 and 1. For nominal data, we first change the data type from object to float and give each group one number, then use the scaling method.

2). Classification method introduces nonlinear into the model, which makes it becoming an effective methodology to predict our data in the AutoGluon model. Linear regression is quite difficult to have an analytical solution defending various factors. The results of classification are better than linear regression, they are large differences between groups, small differences within groups for classification, and differences within groups for equivalent distribution grouping are much smaller than normal distribution classification.

3). Using the AutoGluon model can save scientists several time in searching for methods to fit our research problem. And machine learning can give us another way to think of the data science application in politics. Through the AutoGluon model, we can know the feature sorting which influences GDP per capita and economic growth. After optimization, machine learning methods can extract features very efficiently and give statisticians and economists an intuition about AI applications in this area.

**Acknowledgement**

**References**

“AutoGluon Predictors — AutoGluon Documentation 0.4.0 Documentation.” https://auto.gluon.ai/stable/api/autogluon.predictor.html#autogluon.tabular.TabularPredictor.leaderboard.

“AutoGluon Tasks — AutoGluon Documentation 0.4.0 Documentation.” https://auto.gluon.ai/stable/api/autogluon.task.html.

Baccini, L., & Urpelainen, J. 2014. “International Institutions and Domestic Politics: Can Preferential Trading Agreements Help Leaders Promote Economic Reform?” *Journal of Politics* 76(1): 195–214.

Besley, T., & Reynal-Querol, M. 2011. “Do Democracies Select More Educated Leaders?” *American Political Science Review* 105(3): 552–66.

Besley, T., Montalvo, J. G., & Reynal‐Querol, M. 2011. “Do Educated Leaders Matter?” *Economic Journal* 121(554): F205-227.

Besley, T., Persson, T., & Reynal‐Querol, M. 2016. “Resilient Leaders and Institutional Reform: Theory and Evidence.” *Economica* 83(332): 584–623.

Bolt, J., & Van Zanden, J. L. 2020. “Maddison Style Estimates of the Evolution of the World Economy. A New 2020 Update.” *Maddison-Project Working Paper WP-15.*

“CatBoost - Open-Source Gradient Boosting Library.” https://catboost.ai/.

Debs, A., & Goemans, H. E. 2010. “Regime Type, the Fate of Leaders, and War.” *American Political Science Review* 104(3): 430–45.

Easterly, W., & Pennings, S. 2018. “Shrinking Dictators: How Much Economic Growth Can We Attribute to National Leaders.” *Development Research Institute Working Paper, 94.*

———. 2020. “Leader Value Added: Assessing the Growth Contribution of Individual National Leaders (No. W27153).” *National Bureau of Economic Research*.

Erickson, N., Mueller, J., Shirkov, A., Zhang, H., Larroy, P., Li, M., & Smola, A. 2020. “Autogluon-Tabular: Robust and Accurate Automl for Structured Data.” *arXiv preprint arXiv:2003.06505.*

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., & Hutter, F. 2015. “Efficient and Robust Automated Machine Learning.” *Advances in neural information processing systems, 28.*

Fuhrmann, M., & Horowitz, M. C. 2015. “When Leaders Matter: Rebel Experience and Nuclear Proliferation.” *Journal of Politics* 77(1): 72–87.

Gift, T., & Krcmaric, D. 2017. “Who Democratizes? Western-Educated Leaders and Regime Transitions.” *Journal of Conflict Resolution* 61(3): 671–701.

Goemans, H. E., Gleditsch, K. S., & Chiozza, G. 2009. “Introducing Archigos: A Dataset of Political Leaders.” *journal of peace research* 46(2): 269–83.

———. 2016. “Archigos: A Data Set on Leaders, 1875–2015.” *New York, USA: University of Rochester.*

Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. 2009. 2 Revista Espanola de las Enfermedades del Aparato Digestivo *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: springer.

Horowitz, M. C., & Stam, A. C. 2014. “How Prior Military Experience Influences the Future Militarized Behavior of Leaders.” *International Organization* 68(3): 527–59.

Jones, B. F., & Olken, B. A. 2005. “Do Leaders Matter? National Leadership and Growth since World War II.” *Quarterly Journal of Economics* 120(3): 835–64.

Jones, B. D. (Ed.). 1989. “Leadership and Politics: New Perspectives in Political Science.” *Studies in Government & Public.*

Jordan, M. I., & Mitchell, T. M. 2015. “Machine Learning: Trends, Perspectives, and Prospects.” *Science* 349(6245): 255–60.

LeDell, E., & Poirier, S. 2020. “H2o Automl: Scalable Automatic Machine Learning.” *In Proceedings of the AutoML Workshop at ICML (Vol. 2020)*.

Li, Y., Wang, Z., Xie, Y., Ding, B., Zeng, K., & Zhang, C. 2021. “Automl: From Methodology to Application.” *In Proceedings of the 30th ACM International Conference on Information & Knowledge Management (pp. 4853-4856).*

DeMesquita, B. B., & Siverson, R. M. 1995. “War and the Survival of Political Leaders: A Comparative Study of Regime Types and Political Accountability.” *American Political Science Review* 89(4): 841–55.

Olson, R. S., & Moore, J. H. “TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning.” *In Workshop on automatic machine learning (pp. 66-74). PMLR.*

Thornton, C., Hutter, F., Hoos, H. H., & Leyton-Brown, K. “Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms.” *In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 847-855).*

“Welcome to Fastai | Fastai.” https://docs.fast.ai/.

Wu, C. X., & Wolford, S. 2018. “Leaders, States, and Reputations.” *Journal of Conflict Resolution* 62(10): 2087–2117.

Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. 2021. “Dive Into Deep Learning.” *arXiv preprint arXiv:2106.11342.*