

**Data Mining – DSCI 5240**

***Group 12 Final Project***

# **FIFA'19 PLAYERS ANALYSIS**



**UNDER THE GUIDANCE OF**

***Dr. Robert Getty***

# Table of Contents

1. Introduction to Data Mining	
i. Uses of Data Mining	1
ii. Types of variables in a dataset	1
2. Using SAS enterprise miner for Data Mining	
i. Introduction to SAS	3
ii. Overview of SEMMA	3
iii. Working of SAS Enterprise miner	4
iv. Benefits of SAS Enterprise miner	4
v. Various nodes used in our project	5
3. Project Description	
i. Executive Summary	8
ii. Data description	8
iii. Data Analysis	10
4. Using Linear Regression	
i. Linear regression model	13
ii. Model comparison of regression models	16
iii. Conclusion	16
5. Using Decision Tree	
i. Running a decision tree	18
ii. First split and completing splits	20
iii. Variable importance	20
iv. Conclusion	21
6. Using Neural Networks	
i. Running a neural network	22
ii. Model comparison	24

7. Using Auto Neural Node	
i. Running an Auto Neural node	25
ii. Conclusion	26
8. Model Comparison	27
9. Conclusion	31
10. References and Tools Used	32

## 1. INTRODUCTION TO DATA MINING

---

**Data mining** may be defined as the process of analyzing hidden patterns of data into meaningful information, which is collected and stored in database warehouses, for efficient analysis. One is that the data is largely opportunistic, in the sense that it was not necessarily acquired for the purpose of statistical inference. The algorithms of Data Mining, facilitating business decision making and other information requirements to ultimately reduce costs and increase revenue.

Mining of Data involves effective data collection and warehousing as well as computer processing. It makes use of sophisticated mathematical algorithms for segmenting the data and evaluating the probability of future events. Data Mining is also alternatively referred to as data discovery and knowledge discovery.

### USES OF DATA MINING

Data mining is used for examining raw data, including sales numbers, prices, and customers, to develop better marketing strategies, improve the performance or decrease the costs of running the business. Also, Data mining serves to discover new patterns of behavior among consumers.

Data Mining is used for predictive and descriptive analysis in business:

- (i) The derived pattern in Data Mining is helpful in better understanding of customer behavior, which leads to better & productive future decision.
- (ii) Data Mining is used for finding the hidden facts by approaching the market, which is beneficial for the business but has not yet reached.
- (iii) It is also used for identifying the area of the market, to achieve marketing goals and generate a reasonably good ROI.
- (iv) Data Mining helps in bringing down operational cost, by discovering and defining the potential areas of investment.

### TYPES OF VARIABLES IN A DATASET

A typical data set has many thousands of observations. An observation can represent an entity such as an individual customer, a specific transaction, or a certain household. Variables in the data set contain specific information such as demographic information, sales history, or financial information for each observation. How this information is used depends on the research question of interest.

When discussing types of data, consider the measurement level of each variable. we can generally classify each variable as one of the following:

- Interval — a continuous variable that contains values across a range. For these variables, the mean (or average) is interpretable. Examples include income, temperature, or height.
- Categorical — a classification variable with a finite number of distinct, discrete values. Examples include gender (male or female) and drink size (small, medium, large). Because these variables are noncontinuous, the mean is not interpretable. Categorical data can be grouped in several ways. SAS Enterprise Miner uses the following groupings:

- Unary — a variable that has the same value for every observation in the data set.
- Binary — a variable that has two possible values (for example, gender).
- Nominal — a variable that has more than two levels, but the values of each level have no implied order. Examples are pie flavors such as cherry, apple, and peach.
- Ordinal — a variable that has more than two levels. The values of each level have an implied order. Examples are drink sizes such as small, medium, and large.

Note: Ordinal variables can be treated as interval or nominal variables when we are not interested in the ordering of the levels. However, nominal variables cannot be treated as ordinal variables because, by definition, there is no implied ordering. To obtain a meaningful analysis, we must construct an appropriate data set and specify the correct measurement level for each variable in that data set.

Data mining enables us to discover insights that drive better decision making. With SAS data mining solutions, we can streamline the discovery process to develop models quickly so we can understand key relationships and find the patterns that matter the most.

## 2. USING SAS ENTERPRISE MINER FOR DATA MINING

---

SAS Enterprise Miner is a comprehensive, graphical workbench for data mining. This widely acclaimed and extensive platform provides capabilities to prepare data for predictive analytics, identify the most significant variables, develop models using the most modern data mining and machine-learning algorithms, easily validate the accuracy and fitness of the model(s), and generate assets that allow a simple deployment of analytical models into your operational applications for automated decision making. Powerful data preparation tools address data quality problems, such as missing values and outliers, and help us develop segmentation rules. Interactive data exploration enables users to create dynamic, linked plots to identify relationships within the data. SAS Enterprise Miner provides dozens of advanced statistical and machine-learning algorithms for descriptive and predictive modeling, including clustering, link and market basket analysis, principal component analysis, decision trees, bagging and boosting, Bayesian networks, neural networks, random forests, linear regression, logistic regression, support vector machine, time series data mining and many more. The SAS Enterprise Miner data mining process is driven by a process flow diagram that we can modify, save and share. The drag-and-drop GUI enables business analysts with little statistical expertise to navigate through the data mining process, while the quantitative expert can go behind the scenes to finetune the analytical models

### **INTRODUCTION TO SAS**

**SAS Enterprise Miner** is an advanced analytics data mining tool intended to help users quickly develop descriptive and predictive models through a streamlined data mining process.

Enterprise Miner's graphical interface enables users to logically move through the five-step SAS SEMMA approach: sampling, exploration, modification, modeling, and assessment. Users can build a process flow by selecting the appropriate tab from Enterprise Miner's toolbar, and then dragging and dropping step-specific nodes onto a pallet.

Enterprise Miner supports several algorithms and techniques, including decision trees, time series, Neural Networks, linear and logistic regression, sequence and web path analysis, market basket analysis, and link analysis.

### **OVERVIEW OF SEMMA**

SEMMA is an acronym used to describe the SAS data mining process. It stands for Sample, Explore, Modify, Model, and Assess. SAS Enterprise Miner nodes are arranged on tabs with the same names.

- **Sample** — These nodes identify, merge, partition, and sample input data sets, among other tasks.
- **Explore** — These nodes explore data sets statistically and graphically. These nodes plot the data, obtain descriptive statistics, identify important variables, and perform association analysis, among other tasks.
- **Modify** — These nodes prepare the data for analysis. Examples of the tasks that we can complete for these nodes are creating additional variables, transforming existing variables, identifying outliers, replacing missing values, performing cluster analysis, and analyzing data with self-organizing maps (SOMs) or Kohonen networks.

- **Model** — These nodes fit a predictive model to a target variable. Available models include decision trees, neural networks, least angle regressions, support vector machines, linear regressions, and logistic regressions.
- **Assess** — These nodes compare competing predictive models. They build charts that plot the percentage of respondents, percentage of respondents captures, lift, and profit. SAS Enterprise Miner also includes the HPDM, Utility, Time Series, and Applications tabs for nodes that provide necessary tools but that are not easily categorized on a SEMMA tab. One such example is the SAS Code node. This node enables us to insert custom SAS code into your process flow diagrams. Another example is the Score Code Export node, which exports the files that are necessary for score code deployment in production systems.

## **WORKING OF SAS ENTERPRISE MINER**

In SAS Enterprise Miner, the data mining process is driven by a process flow diagram that we create by dragging nodes from a toolbar that is organized by SEMMA categories and dropping them onto a diagram workspace.

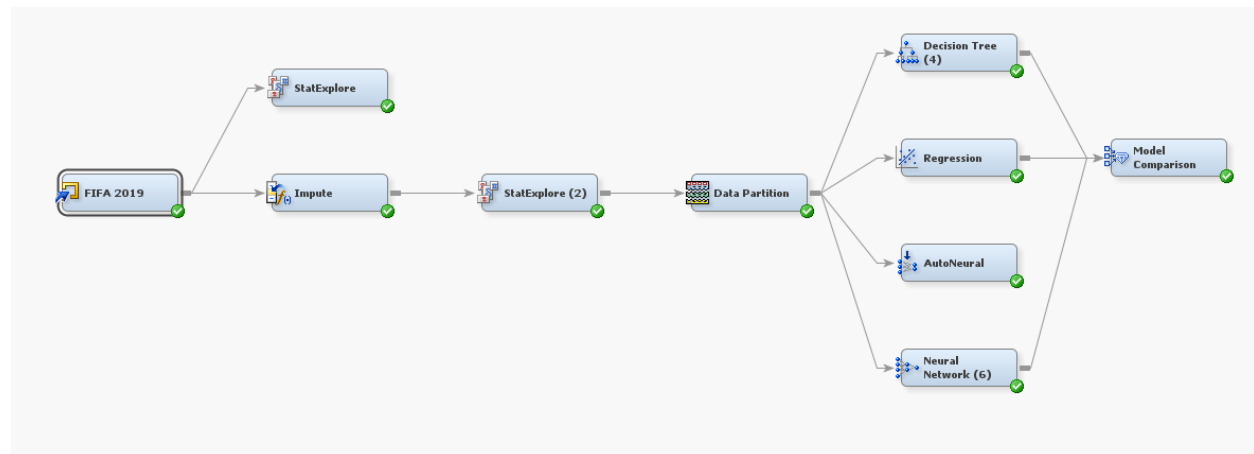


Fig: Example Process Flow Diagram

The graphical user interface (GUI) is designed in such a way that the business analyst who has little statistical expertise can navigate through the data mining methodology, and the quantitative expert can explore each node in depth to fine-tune the analytical process. SAS Enterprise Miner automates the scoring process and supplies complete scoring code for all stages of model development in SAS, C, Java, and PMML. The scoring code can be deployed in a variety of real-time or batch environments within SAS, on the web, or directly in relational databases.

## **BENEFITS OF USING SAS ENTERPRISE MINER**

The benefits of using SAS Enterprise Miner include the following:

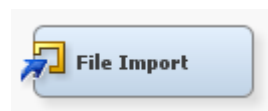
- Support the entire data mining process with a broad set of tools - Regardless of data mining preference or skill level, SAS Enterprise Miner is flexible and addresses complex problems. Going from raw data to accurate, business-driven data mining models becomes a seamless process, enabling the statistical modeling group, business managers, and the IT department to collaborate more efficiently.
- Build more models faster with an easy-to-use GUI- The process flow diagram environment dramatically shortens model development time for both business analysts and statisticians. SAS Enterprise Miner includes an intuitive user interface that incorporates common design principles established for SAS

software and additional navigation tools for moving easily around the workspace. The GUI can be customized for all analysts' needs via flexible, interactive dialog boxes, code editors, and display settings.

- Enhance accuracy of predictions- Innovative algorithms enhance the stability and accuracy of predictions, which can be verified easily by visual model assessment and validation. Both analytical and business users enjoy a common, easy-to-interpret visual view of the data mining process. The process flow diagrams serve as self-documenting templates that can be updated easily or applied to new problems without starting over from scratch.
- Surface business information and easily share results through the unique model repository. Numerous integrated assessment features enable us to compare results of different modeling techniques in both statistical and business terms within a single, easy-to-interpret framework. SAS Enterprise Miner projects support the collaborative sharing of modeling results among quantitative analysts. Models can also be imported into the SAS Model Manager repository for sharing with scoring officers and independent model validation testers.

## **VARIOUS NODES USED IN OUR PROJECT**

### **File Import**



The **File Import** node enables to import data that is stored in external formats into a data source that SAS Enterprise Miner can interpret. The File Import node is located on the Sample tab of the SAS Enterprise Miner toolbar. The File Import node currently can process CSV flat files, JMP tables, Microsoft Excel and Lotus spreadsheet files, Microsoft Access database tables, and DBF, DLM, and DTA files.

### **Data partition**



The **Data Partition** node enables to partition data sets into training, test, and validation data sets. The Data Partition node must be preceded by a node that exports at least one raw, train, or document data table, which is usually an Input Data node or a Sample node. The training data set is used for preliminary model fitting. The validation data set is used to monitor and tune the model weights during estimation and is also used for model assessment. The test data set is an additional holdout data set that we can use for model assessment. This node uses simple random sampling, stratified random sampling, or user-defined partitions to create partitioned data sets.

### **Stat Explore**



The **Stat Explore** node is a multipurpose node that is used to examine variable distributions and statistics in your data sets. We use the Stat Explore node to compute standard univariate statistics, to compute standard bivariate statistics by class target and class segment, and to compute correlation statistics for



interval variables by interval input and target. we can also use the Stat Explore node to reject variables based on target correlation.

## Impute



The **Impute** node enables us to replace missing values for interval variables with the mean, median, midrange, mid-minimum spacing, distribution-based replacement, or use a replacement M-estimator such as Tukey's biweight, Hubers, or Andrew's Wave, or by using a tree-based imputation method. Missing values for class variables can be replaced with the most frequently occurring value, distribution-based replacement, tree-based imputation, or a constant.

## Regression



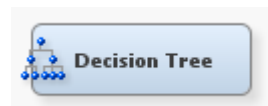
The **Regression** node belongs to the Model category in the SAS data mining process of Sample, Explore, Modify, Model, Assess (SEMMA). The Regression node enables to fit both linear and logistic regression models to your data. We can use continuous, ordinal, and binary target variables. we can use both continuous and discrete variables as inputs. The node supports the stepwise, forward, and backward selection methods. A point-and-click interaction builder enables to create higher-order modeling terms.

## Variable Selection



The **Variable Selection node** tool is located on the Explore tab of the Enterprise Miner tools bar. The Variable Selection node enables to evaluate the importance of input variables in predicting or classifying the target variable. The node uses either an R-square or a Chi-square selection (tree-based) criterion. The R-square criterion removes variables that have large percentages of missing values, and removes class variables that are based on the number of unique values. The variables that are not related to the target are set to a status of rejected. Although rejected variables are passed to subsequent tools in the process flow diagram, these variables are not used as model inputs by modeling nodes such as the Neural Network and Decision Tree tools. If a variable interest is rejected, we can force that variable into the model by reassigning the variable role in any modeling node.

## Decision tree



The **Decision Tree** node is in the Model folder of the SAS Enterprise Miner toolbar. The Decision Tree node enables us to fit decision tree models to your data. The implementation includes features that are found in a variety of popular decision tree algorithms (for example, CHAID, CART). The node supports both

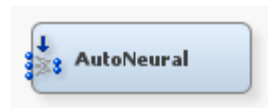
automatic and interactive training. When we run the Decision Tree node in automatic mode, it automatically ranks the input variables based on the strength of their contribution to the tree. This ranking can be used to select variables for use in subsequent modeling. We can override any automatic step with the option to define a splitting rule and prune explicit tools or subtrees. Interactive training enables us to explore and evaluate data splits as we develop them.

## Neural Network



The **Neural Network** node enables to construct, train, and validate multilayer feedforward neural networks. Users can select from several predefined architectures or manually select input, hidden, and target layer functions and options.

## Auto Neural



The **Auto Neural** node can be used to automatically configure a neural network. The Auto Neural node implements a search algorithm to incrementally select activation functions for a variety of multilayer networks. The Auto Neural node conducts limited searches in order to find better network configurations.

## Model Comparison



The **Model Comparison** node belongs to the Assess category in the SAS data mining process of Sample, Explore, Modify, Model, and Assess (SEMMA). The Model Comparison node provides a common framework for comparing models and predictions from any of the modeling tools (such as Regression, Decision Tree, and Neural Network tools). The comparison is based on standard model fit statistics as well as potential expected and actual profits or losses that would result from implementing the model.

:

### 3. PROJECT DESCRIPTION

---

#### **EXECUTIVE SUMMARY**

The dataset can be used as a basis to select players for the upcoming FIFA across the globe by focusing on the below objectives:

1. Is the potential of a player dependent on his age?
2. To create the best model for predicting the potential of a player based on several attributes.

We used Linear Regression to find whether the potential of a player is dependent on his age and other skill sets or not. For predicting the best model among we have created various models varying from Regression to Auto neural and then fed the outputs to Model Comparison. This model comparison will fulfil our second objective.

#### **DATA DESCRIPTION**

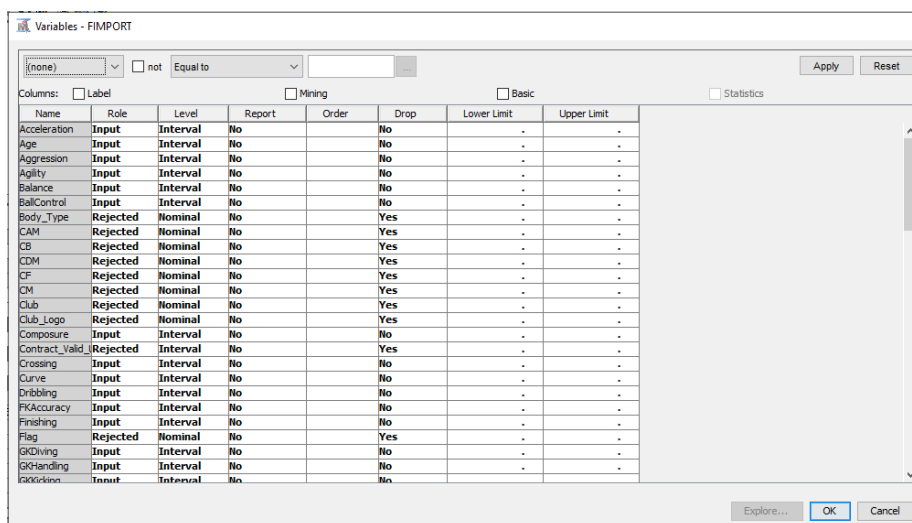
We had chosen a dataset that contains detailed attributes of 18207 players registered in the latest edition of FIFA'19 database. There are 89 attributes which includes some of the personal information of players like age, nationality, name, wage etc., and players skill information like ball control, dribbling, crossing, finishing, GK skills etc. This dataset consists of both nominal and interval variables and there are variables in this dataset which can be used for prediction and decision making.

The main variables used for this project are age, potential, value, wage and all the skills set of the players. Age and potential are used for linear regression model.

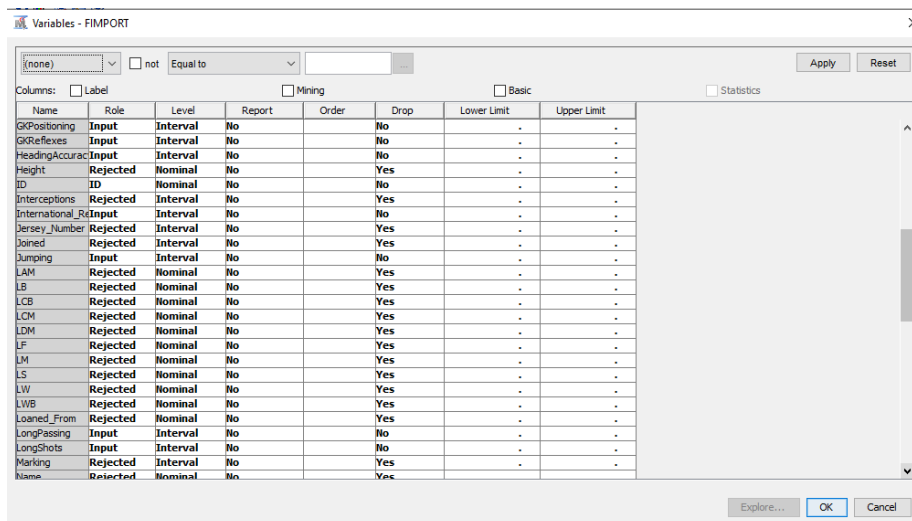
Variable name	Description
ID	Unique id for every player
Name	Name of the player
Age	Age of player
Potential	Potential rating
Value	Current market value
Wage	Yearly earning through the team
Contract Valid Until	Date until which the player signed the contract
LS, ST, RS, LW, LF, CF, RF, RW, LAM, CAM, RAM, LM, LCM, CM, RCM, RM, LWB, LDM, CDM, RDM, RWB, LB, LCB, CB, RCB, RB, Crossing, Finishing, HeadingAccuracy, ShortPassing, Volleys, Dribbling, Curve, FKAccuracy, LongPassing, BallControl, Acceleration, SprintSpeed, Agility, ReactionsBalance, ShotPower, Jumping, Stamina, Strength, LongShots, Aggression, Interceptions, Positioning, Vision, Penalties, Composure, Marking, StandingTackle, SlidingTackle, GK Diving, GK Handling, GK Kicking, GK Positioning, GK Reflexes	Skills set rated on a scale of 1-100

The role of variables acceleration, age, aggression, agility, balance, ball control, composure, crossing, curve, dribbling, FKAccuracy, Finishing, GK Diving, GKHandling, GKKicking, GKPositioning, GKReflexes, Heading Accuracy, International\_reputation, jumping, LongPassing, Longshots, Penalties, Shortpassing, ShotPower, Skill\_moves, SlidingTackle, StandingTackle, Strength, and Value are set as input for the analysis and the rest of the variables are rejected as all other variables are related to player skills. Therefore, we rejected the other variables. This process is called data reduction which is explained in below section. The variable **potential** is selected as target. The role for variable ID is set to ID as it is a unique value for each player in the dataset.

### *Screenshot showing the role of variables:*



Name	Role	Level	Report	Order	Drop	Lower Limit	Upper Limit
Acceleration	Input	Interval	No		No	-	-
Age	Input	Interval	No		No	-	-
Aggression	Input	Interval	No		No	-	-
Agility	Input	Interval	No		No	-	-
Balance	Input	Interval	No		No	-	-
BallControl	Input	Interval	No		No	-	-
Body_Type	Rejected	Nominal	No		Yes	-	-
CAM	Rejected	Nominal	No		Yes	-	-
CB	Rejected	Nominal	No		Yes	-	-
CDM	Rejected	Nominal	No		Yes	-	-
CF	Rejected	Nominal	No		Yes	-	-
CM	Rejected	Nominal	No		Yes	-	-
Club	Rejected	Nominal	No		Yes	-	-
Club_Logo	Rejected	Nominal	No		Yes	-	-
Composure	Input	Interval	No		No	-	-
Contract_Valid	Rejected	Interval	No		Yes	-	-
Crossing	Input	Interval	No		No	-	-
Curve	Input	Interval	No		No	-	-
Dribbling	Input	Interval	No		No	-	-
FKAccuracy	Input	Interval	No		No	-	-
Finishing	Input	Interval	No		No	-	-
Flag	Rejected	Nominal	No		Yes	-	-
GKDivining	Input	Interval	No		No	-	-
GKHandling	Input	Interval	No		No	-	-
GKGoalkeeping	Input	Interval	No		No	-	-



Name	Role	Level	Report	Order	Drop	Lower Limit	Upper Limit
GKPositioning	Input	Interval	No		No	-	-
GKReflexes	Input	Interval	No		No	-	-
HeadingAccuracy	Input	Interval	No		No	-	-
Height	Rejected	Nominal	No		Yes	-	-
ID	ID	Nominal	No		No	-	-
Interceptions	Rejected	Interval	No		Yes	-	-
International_R	Input	Interval	No		No	-	-
Jersey_Number	Rejected	Interval	No		Yes	-	-
Joined	Rejected	Interval	No		Yes	-	-
Jumping	Input	Interval	No		No	-	-
LAM	Rejected	Nominal	No		Yes	-	-
LB	Rejected	Nominal	No		Yes	-	-
LCB	Rejected	Nominal	No		Yes	-	-
LCM	Rejected	Nominal	No		Yes	-	-
LDM	Rejected	Nominal	No		Yes	-	-
LF	Rejected	Nominal	No		Yes	-	-
LM	Rejected	Nominal	No		Yes	-	-
LS	Rejected	Nominal	No		Yes	-	-
LW	Rejected	Nominal	No		Yes	-	-
LWB	Rejected	Nominal	No		Yes	-	-
Loaned_From	Rejected	Nominal	No		Yes	-	-
LongPassing	Input	Interval	No		No	-	-
LongShots	Input	Interval	No		No	-	-
Marking	Rejected	Interval	No		Yes	-	-
Name	Rejected	Nominal	No		Yes	-	-

Variables - FIMPORT

Columns: ☐ Label ☐ Mining ☐ Basic ☐ Statistics

Name	Role	Level	Report	Order	Drop	Lower Limit	Upper Limit
Nationality	Rejected	Nominal	No		Yes	-	-
Overall	Rejected	Interval	No		No	-	-
Penalties	Input	Interval	No		No	-	-
Photo	Rejected	Nominal	No		Yes	-	-
Position	Rejected	Nominal	No		Yes	-	-
Positioning	Rejected	Interval	No		Yes	-	-
Potential	Target	Interval	No		No	-	-
Preferred_Foot	Rejected	Nominal	No		Yes	-	-
RAM	Rejected	Nominal	No		Yes	-	-
RB	Rejected	Nominal	No		Yes	-	-
RCB	Rejected	Nominal	No		Yes	-	-
RCM	Rejected	Nominal	No		Yes	-	-
RDM	Rejected	Nominal	No		Yes	-	-
RF	Rejected	Nominal	No		Yes	-	-
RM	Rejected	Nominal	No		Yes	-	-
RS	Rejected	Nominal	No		Yes	-	-
RW	Rejected	Nominal	No		Yes	-	-
RWB	Rejected	Nominal	No		Yes	-	-
Reactions	Rejected	Interval	No		Yes	-	-
Real_Face	Rejected	Nominal	No		Yes	-	-
ST	Rejected	Nominal	No		Yes	-	-
ShortPassing	Input	Interval	No		No	-	-
ShotPower	Input	Interval	No		No	-	-
Skill_Moves	Input	Interval	No		No	-	-
SidronTackle	Input	Interval	No		No	-	-

Explore... OK Cancel

Special	Rejected	Interval	No		Yes	-	-
SprintSpeed	Rejected	Interval	No		Yes	-	-
Stamina	Rejected	Interval	No		Yes	-	-
StandingTackle	Input	Interval	No		No	-	-
Strength	Input	Interval	No		No	-	-
Value	Input	Interval	No		No	-	-
Vision	Rejected	Interval	No		Yes	-	-
Volleys	Rejected	Interval	No		Yes	-	-
Wage	Rejected	Interval	No		Yes	-	-
Weak_Foot	Rejected	Interval	No		Yes	-	-
Weight	Rejected	Nominal	No		Yes	-	-
Work_Rate	Rejected	Nominal	No		Yes	-	-

Explore... OK Cancel

## DATA ANALYSIS:

### 1) DATA PREPARATION (PREPROCESSING TASKS):

**DATA CLEANING:** It is the process of identifying incomplete data and then replacing them with the appropriate values.

We had selected a dataset first and imported the data into the SAS Enterprise Miner by using the File Import node and connected this node to the stat explore node. The functionality of the Stat Explore node is to analyze the data and find out if there are any missing values or not. So, from the dataset we used we found that there are many missing values for different variables which we have chosen. Hence, in order to clean the data and add the missing values we use the Impute node and connect it to the File Import node.

The Impute node enables us to impute missing values in the input data set. Imputation is necessary to ensure that every observation in the training data is used when we build a regression or neural network model. Tree models can handle missing values directly. But regression and neural network models ignore incomplete observations. It is more appropriate to compare models that are built on the same set of observations. We should perform variable replacement or imputation before fitting any regression or neural network model that we want to compare to a tree model.

The Impute node enables us to specify an imputation method that replaces every missing value with some statistic. By default, interval input variables are replaced by the mean of that variable. Class input variables are replaced by the most frequent value for that variable. In this we used mean as the replacement value.

After this process is done, we need to connect the Impute node to the Stat Explore node and we get to see that there are no missing values in the dataset.

## BEFORE IMPUTING:

From the figure below we can see that there are many missing values which is highlighted.

Variable	Role	Mean	Standard Deviation	Non Missing	Missing	Minimum	Median	Maximum	Skewness	Kurtosis
Acceleration	INPUT	64.61408	14.92778	18159	48	12	67	97	-0.8153	0.469644
Age	INPUT	25.12221	4.669943	18207	0	16	25	45	0.391764	-0.45951
Aggression	INPUT	55.86899	17.36797	18159	48	11	59	95	-0.43217	-0.63114
Agility	INPUT	63.50361	14.76605	18159	48	14	66	96	-0.59942	-0.0686
Balance	INPUT	63.96657	14.13617	18159	48	16	66	96	-0.58327	0.093969
BallControl	INPUT	58.36946	16.6866	18159	48	5	63	96	-1.26786	1.030129
Composure	INPUT	58.64827	11.43613	18159	48	3	60	96	-0.38103	0.214397
Crossing	INPUT	49.73418	18.36452	18159	48	5	54	93	-0.59448	-0.54331
Curve	INPUT	47.17082	18.39526	18159	48	6	48	94	-0.23819	-0.74397
Dribbling	INPUT	55.371	18.91037	18159	48	4	61	97	-1.0842	0.337456
FKAccuracy	INPUT	42.86315	17.47876	18159	48	3	41	94	0.102117	-0.71834
Finishing	INPUT	45.55091	19.52582	18159	48	2	49	95	-0.30088	-0.97261
GKDividing	INPUT	16.61622	17.69535	18159	48	1	11	90	2.441015	4.445862
GKHandling	INPUT	16.3916	16.9069	18159	48	1	11	92	2.433494	4.441784
GKkicking	INPUT	16.23206	16.50286	18159	48	1	11	91	2.430576	4.46266
GKPositioning	INPUT	16.3889	17.03467	18159	48	1	11	90	2.4638	4.633651
GKReflexes	INPUT	16.71089	17.95512	18159	48	1	11	94	2.452638	4.510532
HeadingAccuracy	INPUT	52.29814	17.37991	18159	48	4	56	94	-0.88513	0.292507
International_Reputation	INPUT	1.113222	0.394031	18159	48	1	1	5	4.06042	18.96098
Jumping	INPUT	65.08943	11.82004	18159	48	15	66	95	-0.45432	0.324853
LongPassing	INPUT	52.71193	15.32787	18159	48	9	56	93	-0.59564	-0.37654
LongShots	INPUT	47.10997	19.26052	18159	48	3	51	94	-0.43227	-0.82051
Penalties	INPUT	48.5486	15.70405	18159	48	5	49	92	-0.35497	-0.34735
ShortPassing	INPUT	58.68671	14.6995	18159	48	7	62	93	-1.10032	0.788778
ShotPower	INPUT	55.46005	17.23796	18159	48	2	59	95	-0.68047	-0.33517
Skill_Moves	INPUT	2.361308	0.756164	18159	48	1	2	5	0.149898	-0.0876
SlidingTackle	INPUT	45.66144	21.28913	18159	48	3	52	91	-0.26656	-1.36351
StandingTackle	INPUT	47.69784	21.664	18159	48	2	55	93	-0.33661	-1.3136
Strength	INPUT	65.31197	12.557	18159	48	17	67	97	-0.46836	0.057389
Value	INPUT	2287046	5543946	18207	0	0	675000	1.18E8	7.220947	79.29853
Potential	TARGET	71.3073	6.136496	18207	0	48	71	95	0.266154	0.035826

## AFTER IMPUTNG:

After connecting the impute node there are no missing values as shown in the figure below.

Variable	Role	Mean	Standard Deviation	Non Missing	Missing	Minimum	Median	Maximum	Skewness	Kurtosis
Age	INPUT	25.10912	4.706023	10924	0	16	25	44	0.400649	-0.48306
IMP_Acceleration	INPUT	64.41146	15.01666	10924	0	12	67	97	-0.84262	0.470798
IMP_Aggression	INPUT	55.64647	17.37845	10924	0	11	58	94	-0.42705	-0.64177
IMP_Agility	INPUT	63.34619	14.7598	10924	0	14	66	95	-0.61519	-0.06017
IMP_Balance	INPUT	63.85385	14.15747	10924	0	16	66	96	-0.59034	0.120003
IMP_BallControl	INPUT	58.15505	16.76848	10924	0	8	63	96	-1.26056	0.973288
IMP_Composure	INPUT	58.60727	11.49658	10924	0	12	59	96	-0.36162	0.158589
IMP_Crossing	INPUT	49.56871	18.37953	10924	0	6	54	93	-0.58987	-0.55149
IMP_Curve	INPUT	46.9776	18.3488	10924	0	6	48	94	-0.24168	-0.75035
IMP_Dribbling	INPUT	55.1433	18.9486	10924	0	4	61	97	-1.08063	0.293678
IMP_FKAccuracy	INPUT	42.74048	17.45612	10924	0	3	41	94	0.098006	-0.71163
IMP_Finishing	INPUT	45.45029	19.55089	10924	0	2	48	95	-0.29509	-0.96475
IMP_GKDividing	INPUT	16.78674	17.82699	10924	0	1	11	90	2.410307	4.297184
IMP_GKHandling	INPUT	16.49601	17.06425	10924	0	1	11	91	2.399108	4.263946
IMP_GKkicking	INPUT	16.35702	16.62244	10924	0	1	11	91	2.397529	4.301861
IMP_GKPositioning	INPUT	16.51446	17.15036	10924	0	1	11	88	2.429624	4.458458
IMP_GKReflexes	INPUT	16.82971	18.07389	10924	0	1	11	94	2.421999	4.363676
IMP_HeadingAccuracy	INPUT	52.16368	17.45308	10924	0	4	55	94	-0.87389	0.274171
IMP_International_Reputation	INPUT	1.111081	0.390962	10924	0	1	1	5	4.113668	19.36996
IMP_Jumping	INPUT	65.02883	11.78729	10924	0	22	66	94	-0.45763	0.339736
IMP_LongPassing	INPUT	52.54292	15.37588	10924	0	9	56	93	-0.59227	-0.37597
IMP_LongShots	INPUT	46.90801	19.2636	10924	0	3	51	94	-0.41857	-0.82549
IMP_Penalties	INPUT	48.4883	15.76241	10924	0	8	49	92	-0.35383	-0.35118
IMP_ShortPassing	INPUT	58.52134	14.73209	10924	0	11	62	93	-1.08432	0.720683
IMP_ShotPower	INPUT	55.31846	17.261	10924	0	3	59	94	-0.67271	-0.35432
IMP_Skill_Moves	INPUT	2.35454	0.754053	10924	0	1	2	5	0.122729	-0.16844
IMP_SlidingTackle	INPUT	45.49885	21.24352	10924	0	3	52	90	-0.25838	-1.36113
IMP_StandingTackle	INPUT	47.51896	21.61243	10924	0	6	55	91	-0.33159	-1.30348
IMP_Strength	INPUT	65.23777	12.50181	10924	0	17	66	97	-0.44744	0.06824
Value	INPUT	2250601	5439361	10924	0	0	675000	1.1E8	7.235781	78.69677
Potential	TARGET	71.27691	6.160888	10924	0	48	71	95	0.231749	0.03559

**DATA INTEGRATION:** It involves combining data residing in different sources and providing users with a unified view of them. Also, it ensures that the incorporation of data from multiple sources has not introduced inconsistencies into the data.

As each row in the data represents each player records with a unique id, we cannot apply data integration.

**DATA REDUCTION:** Generally, there will be datasets, where most of the data will not be used. So, we do data reduction in order to save memory and to execute faster by identifying a smaller subset of the data which can produce the same (or similar) analytical results.

In the dataset used here as well, there are many variables like the jersey number, country of the player, flag, image, and so on which are not the attributes to our objectives hence we have dropped those variables from the dataset by applying **Manual Feature Selection** Process.

## 2) DATA PARTITIONING:

In our diagram workspace, we selected the **Data Partition** node. The default **Partitioning Method** for the **Data Partition** node is stratified. The available methods for partitioning are as follows:

Simple Random — Every observation in the data set has the sample probability to be selected for a specific partition.

Stratified — When we use stratified partitioning, specific variables form strata (or subgroups) of the total population. Within each stratum, all observations have an equal probability of being assigned to one of the partitioned data sets.

Cluster — When we use cluster partitioning, distinct values of the cluster variable are assigned to the various partitions using simple random partitioning.

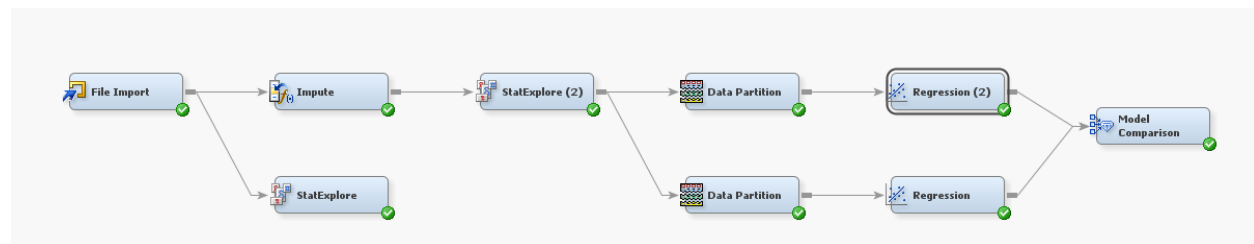
## 4. USING LINEAR REGRESSION

### LINEAR REGRESSION MODEL:

Linear regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). It looks for statistical relationship but not deterministic relationship.

Data Partitioning is one of the important tasks in data mining as it allows us to assess the model and we used the default partitioning method in our project. Here, we have used many combinations and compared them with each other. In the default mode in which the data set allocation as shown has a F value of 2647.99 and R-square as 0.8361 when Linear Regression is done. Whereas, when we have allocated more data to training and less to validation and testing the F value and R-square values and also the number of variables which contributes to the model have increased which proves that more the data is allocated to training, the more we can assess the model.

*Screenshot showing the diagram of the nodes:*

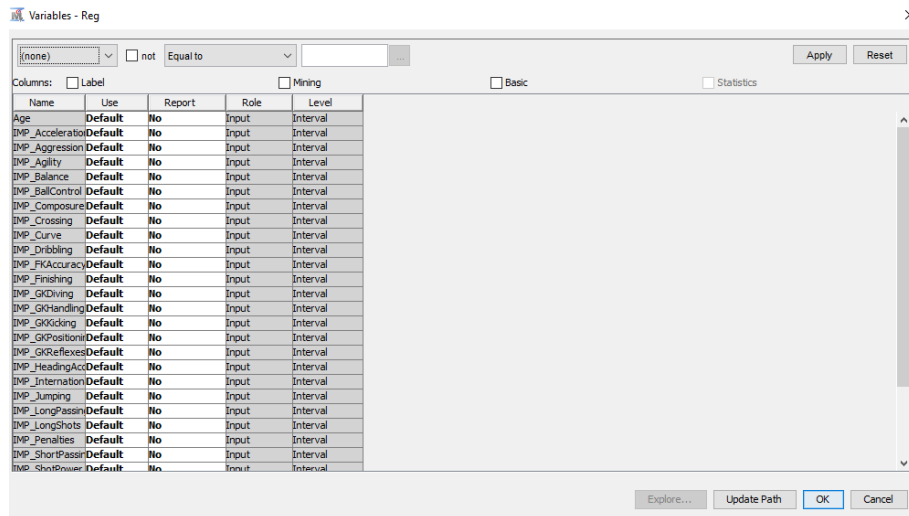


The objective of our project is to determine if the potential of the player is affected by age and many other skills like Age, Crossing, Finishing, HeadingAccuracy, ShortPassing, Volleys, Dribbling, Curve, FKAccuracy, LongPassing, BallControl, Acceleration, SprintSpeed, Agility, ReactionsBalance, ShotPower, Jumping, Stamina, Strength, LongShots, Aggression, Interceptions, Positioning, Vision, Penalties, Composure, Marking, StandingTackle, SlidingTackle, GK Diving, GK Handling, GK Kicking, GK Positioning, GK Reflexes.

In this we have used Stepwise linear regression as it is like forward selection except that at each step, we consider dropping predictors that are not statistically significant, as in backward elimination. It consists of a variable selection method where various combinations of variables are tested together. The “first step” will identify the “best” one-variable model. Subsequent steps will identify the “best” two-variable, three-variable, etc. models. Hence, we have chosen the variables and performed the stepwise linear regression.

**A) Regression Model 1:** The variables are set as shown in figure below:





For this we allocated the data to training, validation and test as 60, 30, 10 respectively. Here we selected all the variables and set it to default and potential of the player to yes so that we can predict whether all the variables are dependent on the potential of the player or not. We want to prove this by allocating the data as above.

Data Set Allocations	
Training	60.0
Validation	30.0
Test	10.0
Report	

**Screenshot showing the results:**

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	22	280766	12762	1039.50	<.0001
Error	10901	133833	12.277120		
Corrected Total	10923	414599			

Model Fit Statistics			
R-Square	0.6772	Adj R-Sq	0.6765
AIC	27417.4990	BIC	27419.5918
SBC	27585.3695	C(p)	23.9982

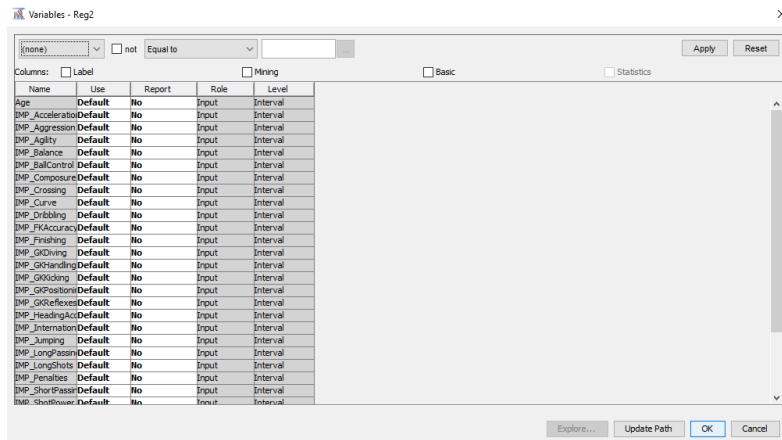
  

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	t Value	Pr >  t
Intercept	1	47.3651	0.4230	111.99	<.0001
Age	1	-0.8526	0.00895	-95.24	<.0001
IMP_Acceleration	1	0.0246	0.00393	6.27	<.0001
IMP_Aggression	1	0.0118	0.00358	3.29	0.0010
IMP_Balance	1	-0.0111	0.00382	-2.91	0.0036
IMP_BallControl	1	0.1552	0.00764	20.32	<.0001
IMP_Composure	1	0.1580	0.00516	30.61	<.0001
IMP_Crossing	1	-0.0206	0.00404	-5.10	<.0001
IMP_GKDividing	1	0.0738	0.0101	7.33	<.0001
IMP_GKHandling	1	0.0962	0.0102	9.43	<.0001
IMP_GKKicking	1	0.0343	0.00942	3.65	0.0003
IMP_GKPositioning	1	0.1029	0.0101	10.22	<.0001
IMP_GKReflexes	1	0.0796	0.00996	7.99	<.0001

The entire regression model is statistically significant here because for a model to be significant the f value associated with its Pr value must be less than 0.05. In this case it is less than 0.0001 which means that

the model is statistically significant. As we used step wise regression, it selected the variables automatically which are significant whose Pr value is less than 0.05. The R-Square value and its Adjusted R-Square value is 0.6772 and 0.6765 respectively.

**B) Regression model 2:** The variables are set as shown in figure below:



In this regression, we allocated the data to training, validation and test as 50, 30, 20 respectively. Because, a model performs well when more data is allocated in training the data.

Data Set Allocations	
Training	50.0
Validation	30.0
Test	20.0

**Screenshot showing the results:**

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	22	233948	10634	869.14	<.0001
Error	9081	111107	12.235067		
Corrected Total	9103	345055			

Model Fit Statistics			
R-Square	0.6780	Adj R-Sq	0.6772
AIC	22822.1744	BIC	22824.2984
SBC	22985.8532	C(p)	21.5134

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	t Value	Pr >  t
Intercept	1	47.2477	0.4616	102.36	<.0001
Age	1	-0.8534	0.00980	-87.07	<.0001
IMP_Acceleration	1	0.0263	0.00429	6.12	<.0001
IMP_Aggression	1	0.0113	0.00392	2.89	0.0038
IMP_Balance	1	-0.0125	0.00418	-2.99	0.0028
IMP_BallControl	1	0.1525	0.00836	18.23	<.0001
IMP_Composure	1	0.1574	0.00563	27.97	<.0001
IMP_Crossing	1	-0.0213	0.00440	-4.84	<.0001
IMP_GKDiving	1	0.0687	0.0110	6.26	<.0001
IMP_GKHandling	1	0.1098	0.0112	9.83	<.0001
IMP_GKicking	1	0.0332	0.0103	3.21	0.0013
IMP_GKPositioning	1	0.1018	0.0110	9.28	<.0001
IMP_GKReflexes	1	0.0736	0.0109	6.75	<.0001

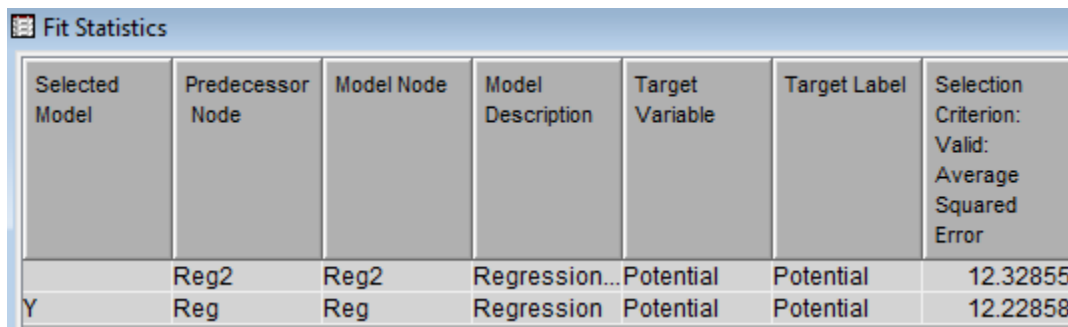
The entire regression model is statistically significant here because for a model to be significant the f value associated with its Pr value must be less than 0.05. In this case it is less than 0.0001 which means that the

model is statistically significant. As we used step wise regression, it selected the variables automatically which are significant whose Pr value is less than 0.05. The R-Square value and its Adjacent R-Square value is 0.6780 and 0.6772 respectively.

### **MODEL COMPARISION OF REGRESSION MODELS**

Now from the above 2 regression models we need to find out the best model among the two. We can do this by using the model comparison node. The use of the model comparison node is to compare the models to which the node is connected, and predictions can be done accordingly. In this case, we compared the two regression models used before and we compared them to find the best one among the two. The model comparison criteria is based on Average Square Error.

*Screenshot showing the fit statistics:*



Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Valid: Average Squared Error
	Reg2	Reg2	Regression...	Potential	Potential	12.32855
Y	Reg	Reg	Regression	Potential	Potential	12.22858

### **CONCLUSION**

From the above figure we can see that the value of average squared error for both the regression models are 12.32 and 12.22, respectively. The model which has the least average squared error is the best model. So, in this case though both the values are almost similar, the first linear regression(**Regression (2)**) model has the lowest value compared to that of the second (**Regression**) and because of this, the linear regression model with the data partition as 60,30, and 10 for train, validation and test respectively is the best among the two models.

#### Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr >  t
Intercept	1	47.3651	0.4230	111.99	<.0001
Age	1	-0.8526	0.00895	-95.24	<.0001
IMP_Acceleration	1	0.0246	0.00393	6.27	<.0001
IMP_Aggression	1	0.0118	0.00358	3.29	0.0010
IMP_Balance	1	-0.0111	0.00382	-2.91	0.0036
IMP_BallControl	1	0.1552	0.00764	20.32	<.0001
IMP_Composure	1	0.1580	0.00516	30.61	<.0001
IMP_Crossing	1	-0.0206	0.00404	-5.10	<.0001
IMP_GKDividing	1	0.0738	0.0101	7.33	<.0001
IMP_GKHandling	1	0.0962	0.0102	9.43	<.0001
IMP_GKKicking	1	0.0343	0.00942	3.65	0.0003
IMP_GKPositioning	1	0.1029	0.0101	10.22	<.0001
IMP_GKReflexes	1	0.0796	0.00996	7.99	<.0001

From the Analysis of Maximum Likelihood estimates we can conclude that the Age variable and other variables are significant in predicting the potential of the player. The first objective of the player which is whether the potential is dependent on age or not can be concluded by this regression model. From the above screenshot we can conclude that **as the age variable increase the potential of the player decreases**. As the variable's acceleration, ball control and aggression values increase potential of the players increases.

## 5. USING DECISION TREE

---

### RUNNING A DECISION TREE

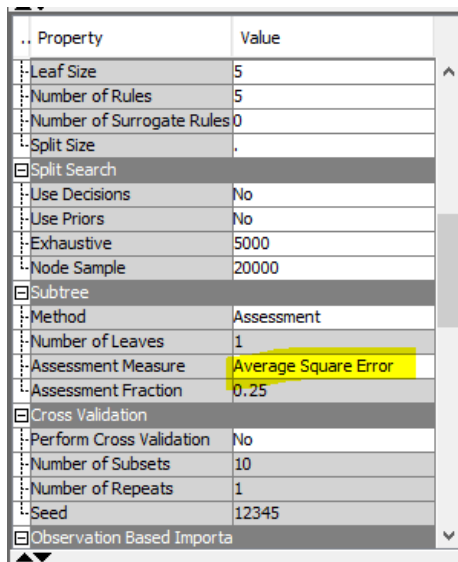
Now that we have modeled and compared two linear regression models with different data partitioning, we want to fit a tree model to determine whether it will make better predictions. On the **Model** tab, we dragged two nodes (**Decision Tree**, **Decision Tree (2)**) to our diagram workspace. We connected the **Data Partition** node to the **Decision Tree** node. As Tree models handle missing values differently than regression models, so we have directly connected the Data Partition node to the Decision Tree node. We also added another 2 decision tree nodes (**Decision Tree (3)**, **Decision Tree (4)**) to Impute node so that we can compare these models with one another.

All the below nodes have assessment measure set to Average Square Error. Other than that, below are the properties set to individual tree nodes:

**Decision Tree and Decision Tree (3)** have maximum branch set to 2

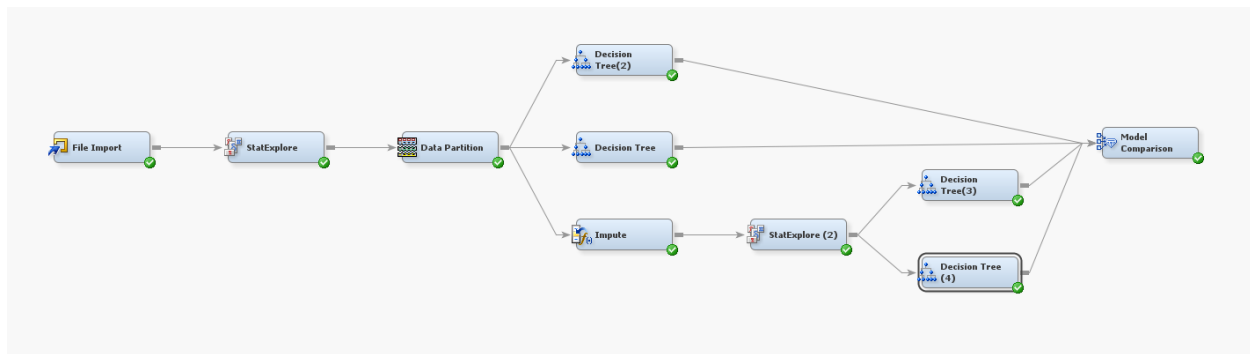
**Decision Tree (2) and Decision Tree (4)** have maximum branch set to 3

*Screenshot showing the properties window for decision tree:*



Property	Value
Leaf Size	5
Number of Rules	5
Number of Surrogate Rules	0
Split Size	.
<input checked="" type="checkbox"/> Split Search	
Use Decisions	No
Use Priors	No
Exhaustive	5000
Node Sample	20000
<input checked="" type="checkbox"/> Subtree	
Method	Assessment
Number of Leaves	1
Assessment Measure	Average Square Error
Assessment Fraction	0.25
<input checked="" type="checkbox"/> Cross Validation	
Perform Cross Validation	No
Number of Subsets	10
Number of Repeats	1
Seed	12345
<input checked="" type="checkbox"/> Observation Based Importa	

*Screenshot showing the diagram of the nodes:*

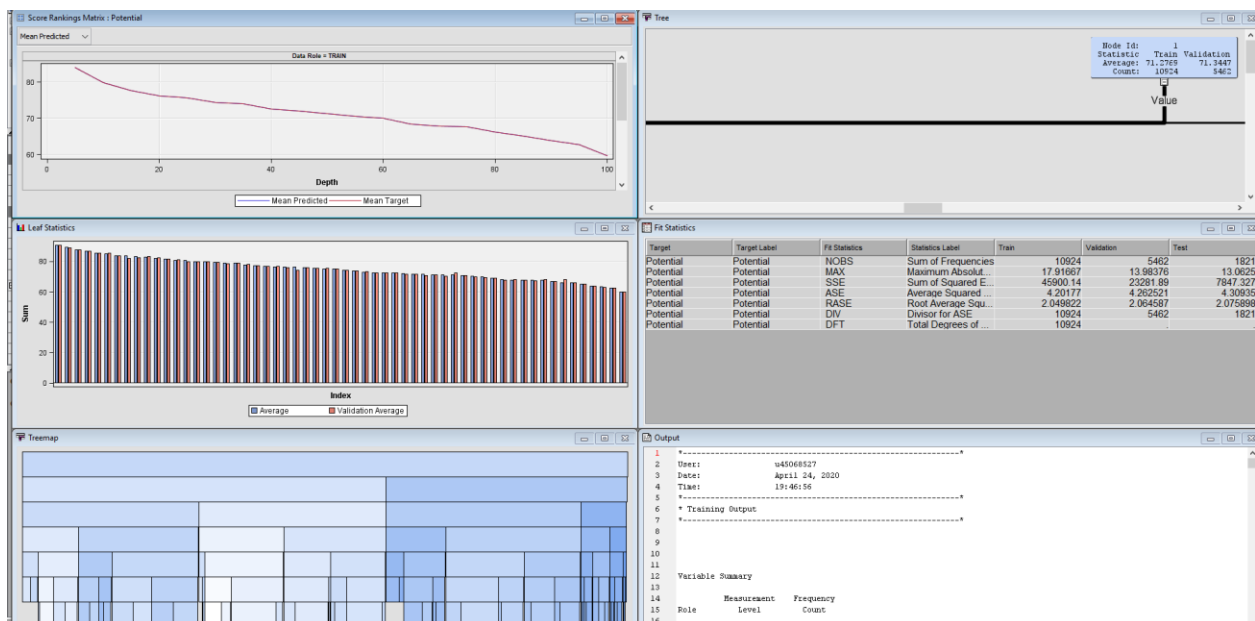


Right-click the **Model Comparison node** and click Run. In the Confirmation window, click Yes. In the Run Status window, click Results.

### Exploring the tree model results

In our diagram workspace, when we right-click the Decision Tree node and click Results, the results are as shown:

*Screenshot showing the results:*



The Tree Map and Tree windows present two different views of the decision tree. The Tree window displays a standard decision tree, where the node splits are provided on the links between the nodes. Inside the nodes, we can find the node ID, the training and validation classification rates, and the number of observations in the node for each partition. The Tree Map window uses the width of each node to visually display the percentage of observations in each node on that row. Selecting a node in one window selects the corresponding node in the other window.

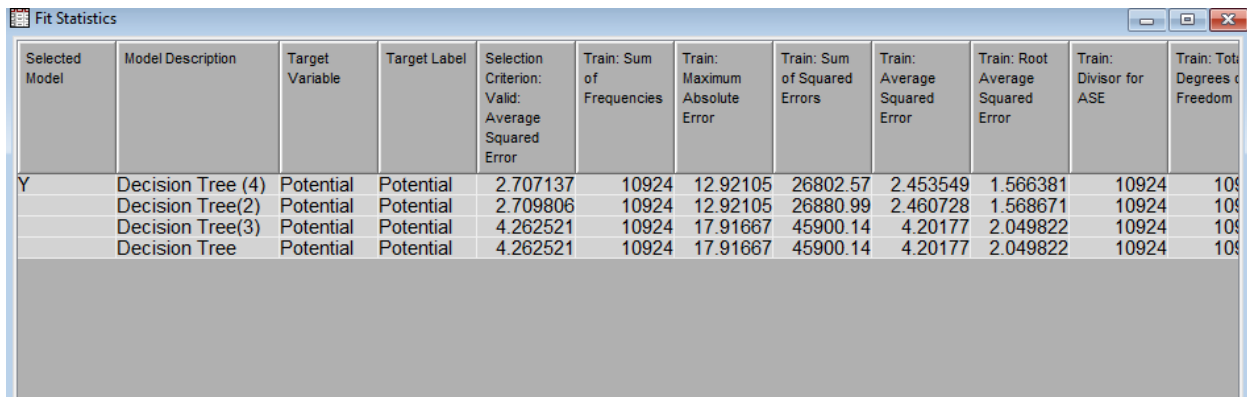
The color of each node in the decision tree is based on the target variable. The Leaf Statistics window displays the training and validation classification rates for each leaf node in the tree model. Leaf nodes are

terminal nodes in the decision tree. Select a bar in the Leaf Statistics window to select the corresponding leaf node in the Tree and Tree Map windows.

The Fit Statistics window displays several computed statistics for the tree model. These are computed for all partitions of the training data.

When we ran the above nodes. The Fit Statistics window confirms that the **Decision Tree (4)** model with 3 maximum branches is the better model based on the selection criteria of the **Model Comparison** node.

*Screenshot showing the Fit Statistics:*

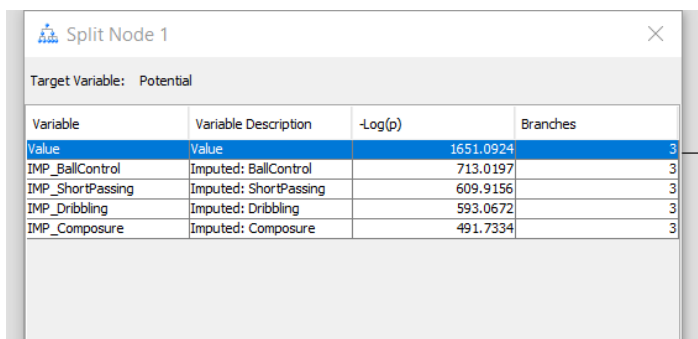


Selected Model	Model Description	Target Variable	Target Label	Selection Criterion: Valid: Average Squared Error	Train: Sum of Frequencies	Train: Maximum Absolute Error	Train: Sum of Squared Errors	Train: Average Squared Error	Train: Root Average Squared Error	Train: Divisor for ASE	Train: Total Degrees of Freedom
Y	Decision Tree (4)	Potential	Potential	2.707137	10924	12.92105	26802.57	2.453549	1.566381	10924	10924
	Decision Tree(2)	Potential	Potential	2.709806	10924	12.92105	26880.99	2.460728	1.568671	10924	10924
	Decision Tree(3)	Potential	Potential	4.262521	10924	17.91667	45900.14	4.20177	2.049822	10924	10924
	Decision Tree	Potential	Potential	4.262521	10924	17.91667	45900.14	4.20177	2.049822	10924	10924

## FIRST SPLIT AND COMPLETING SPLITS

In order to know the variable that was used for the first split and the completing splits for the first split, we have selected the interactive ellipsis (...) from the decision Tree node's Properties panel. The SAS Enterprise Miner Tree window opens. Right clicking on first top node and selecting split node we got the as below

*Screenshot showing the first split and completing splits:*



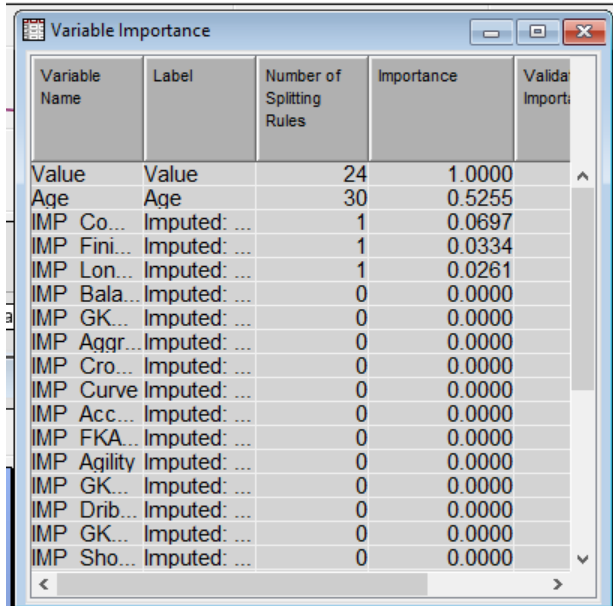
Variable	Variable Description	-Log(p)	Branches
Value	Value	1651.0924	3
IMP_BallControl	Imputed: BallControl	713.0197	3
IMP_ShortPassing	Imputed: ShortPassing	609.9156	3
IMP_Dribbling	Imputed: Dribbling	593.0672	3
IMP_Composure	Imputed: Composure	491.7334	3

This shows that the first split node is Value and the completing splits for the first split are IMP\_BallControl, IMP\_ShortPassing, IMP\_Dribbling, IMP\_Composure.

## VARIABLE IMPORTANCE

In the results window when we click view>>model>>variable importance the following window appears:

*Screenshot showing the variable importance:*



Variable Name	Label	Number of Splitting Rules	Importance	Valida Import
Value	Value	24	1.0000	
Age	Age	30	0.5255	
IMP Co...	Imputed: ...	1	0.0697	
IMP Fini...	Imputed: ...	1	0.0334	
IMP Lon...	Imputed: ...	1	0.0261	
IMP Bala...	Imputed: ...	0	0.0000	
IMP GK...	Imputed: ...	0	0.0000	
IMP Aggr...	Imputed: ...	0	0.0000	
IMP Cro...	Imputed: ...	0	0.0000	
IMP Curve	Imputed: ...	0	0.0000	
IMP Acc...	Imputed: ...	0	0.0000	
IMP FKA...	Imputed: ...	0	0.0000	
IMP Agility	Imputed: ...	0	0.0000	
IMP GK...	Imputed: ...	0	0.0000	
IMP Drib...	Imputed: ...	0	0.0000	
IMP GK...	Imputed: ...	0	0.0000	
IMP Sho...	Imputed: ...	0	0.0000	

## **CONCLUSION**

From the above screenshot we can say that Age variable has an importance of 0.5255. By this we can conclude that Age has more than 50% importance in deciding the potential of the player.

We can also conclude that out of the 4 decision tree nodes, the decision Tree (4) node with maximum number of branches 3 outperformed the other three Tree nodes. Hence, we are choosing this tree node to compare with all the other nodes (i.e., Regression, Auto Neural, Neural Network).



## 6. USING NEURAL NETWORKS

---

### RUNNING A NEURAL NETWORK

Neural networks are a class of parametric models that can accommodate a wider variety of nonlinear relationships between a set of predictors and a target variable than can logistic regression. Building a neural network model involves two main phases. First, we must define the network configuration. We can think of this step as defining the structure of the model that we want to use. Then, we iteratively train the model.

A neural network model will be more complicated to explain to the management of your organization than a regression or a decision tree. However, we know that the management would prefer a stronger predictive model, even if it is more complicated. So, we decide to run a neural network model, which we will compare to the other models later in the example.

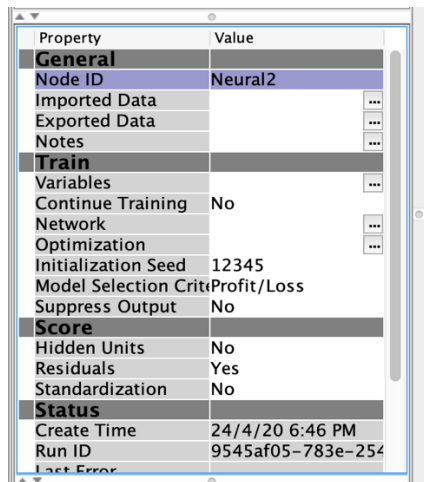
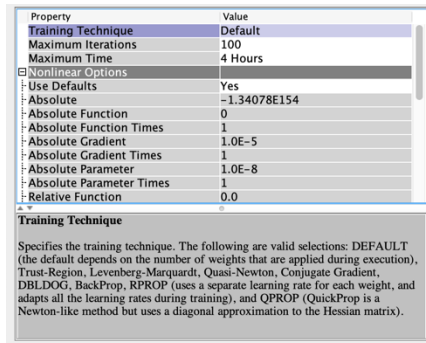
Because neural networks are so flexible, SAS Enterprise Miner has two nodes that fit neural network models: The Neural Network node and the AutoNeural node. The Neural Network node trains a specific neural network configuration; this node is best used when we know a lot about the structure of the model that we want to define. The AutoNeural node (explained in later section) searches over several network configurations to find one that best describes the relationship in a data set and then trains that network.

As we have observed the comparison between different decision models now, we will be comparing different neural networks. One of the key strengths of neural networks is that they have high predictive performance. Their structure supports capturing very complex relationships between predictors and a response, which is often not possible with other classifiers.

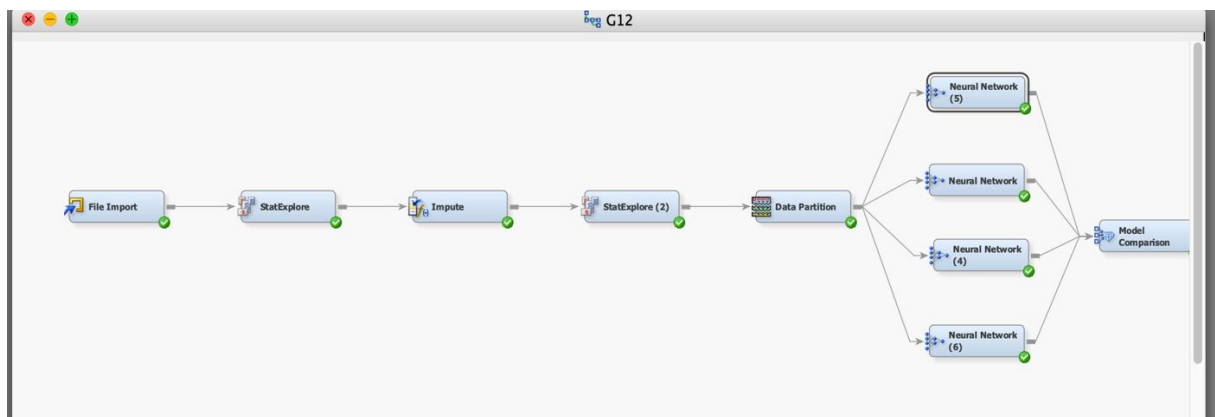
From the Model tab we have dragged Neural Network node. We know that neural networks train by adjusting neuron input weights based on the network performance on hidden units. Neural networks are networks in which there is an input layer consisting of nodes that simply accept the input values and successive layers of nodes that receive input from the previous layers. The outputs of nodes in a layer are inputs to nodes in the next layer. The last layer is called the output layer. Layers between the input and output layers are known as hidden layers. Here in this model we have used four Neural network nodes. (**Neural Network**, **Neural Network (4)**, **Neural Network (5)**, **Neural Network (6)**). We have used the number within the brackets to specify the number of hidden units used **Neural Network** has 3 hidden layers which is a default value for any neural network node. **Neural Network (4)**, **Neural Network (5)**, **Neural Network (6)** has 4,5,6 hidden layers respectively

All the nodes have Maximum Iterations set to 100. Other than that, below are the properties of individual Neural network nodes:

### Screenshots showing the properties window for Neural Network mode

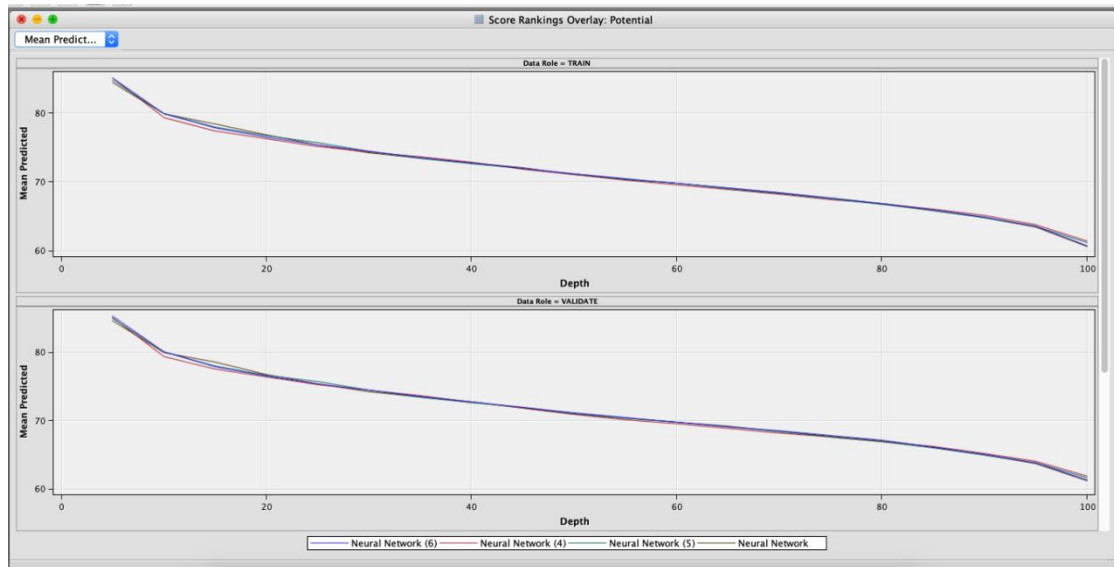


### Screenshot showing the diagram of the nodes:



After placing all the required neural network nodes, we have used Model Comparison node from Assess tab to compare different neural networks.

### Screenshot showing the Score Rankings Overlay: Potential of the neural networks:



In the above screenshot we can observe that all the networks are having a close overlay of score rankings

### MODEL COMPARISON

In order to compare the models, we used average squared error as selection criteria.

Average squared error is measure of how close a fitted line is to data points. For every data point, we take the distance vertically from the point to the corresponding y value on the curve fit (the error) and square the value. The less the average squared error better the performance of the model.

### Screenshot showing the Fit Statistics:

Predecessor Node	Model Node	Model Description	Target	Target Label	Selection Criterion: Valid: Average Squared Error	Train: Total Degrees of Freedom	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom	Train: Number of Estimated Weights	Train: Akaike's Information Criterion	Train: Schwarz's Bayesian Criterion	Train: Average Squared Error	Train: Maximum Absolute Error	Train: Divisor for ASE	Train: Sum of Frequencies	Train: Root Average Squared Error	Train: Sum of Squared Errors	Train: Sum of Case Weights Times Fr
Neural4	Neural4	Neural ... Potential	Potential	Potential	4.68091	10924	10731	193	193	16506....	17914....	4.3739...	14.328...	10924	10924	2.0913...	47780....	109
Neural4	Neural4	Neural ... Potential	Potential	Potential	4.8095...	10924	10827	97	97	16575....	17283....	4.4799...	18.342...	10924	10924	2.1165...	48938....	109
Neural2	Neural2	Neural ... Potential	Potential	Potential	4.8472...	10924	10763	161	161	16995....	18171....	4.6013...	13.353...	10924	10924	2.1450...	50265.5	109
Neural3	Neural3	Neural ... Potential	Potential	Potential	6.0095...	10924	10795	129	129	19640....	20582....	5.8961...	15.082...	10924	10924	2.4282...	64409....	109

Selected Model	Model Node	Model Description	Valid: Average Squared Error	Train: Average Squared Error	Train: Misclassification Rate
Y	Neural4	Neural Network (6)	4.68091	4.37394	-
	Neural1	Neural Network	4.80958	4.47992	-
	Neural2	Neural Network (5)	4.84729	4.60138	-
	Neural3	Neural Network (4)	6.00954	5.89617	-

### CONCLUSION

In the above picture we can observe that for Neural Network with 6 hidden layers the Average squared error is less which means that it has better performance than other networks. Neural networks with 3,5 hidden layers are close in performance whereas the one with 4 hidden layers has the lowest performance of all. Therefore, our network with 6 hidden layers proves to be a better fit to the input data.

## 7. USING AUTO NEURAL NODE

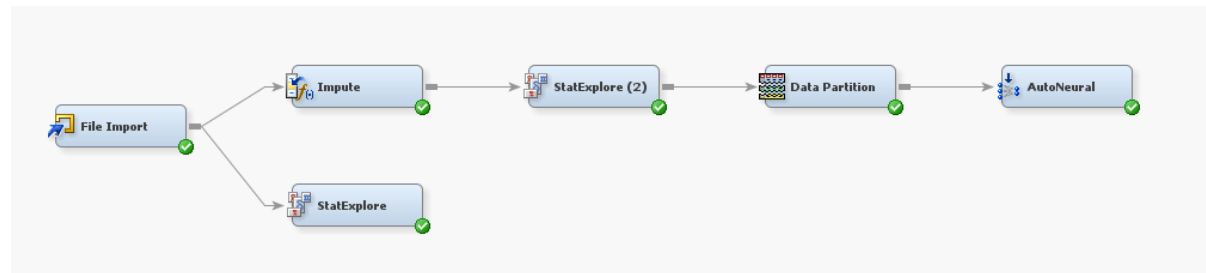
---

### RUNNING AN AUTO NEURAL NODE

The difference between a Neural node and AutoNeural node is that, the Neural Network node trains a specific neural network configuration; this node is best used when we know a lot about the structure of the model that we want to define. The AutoNeural node searches over several network configurations to find one that best describes the relationship in a data set and then trains that network.

From the Model tab on the Toolbar, we dragged the AutoNeural Network node icon into the Diagram Workspace and connected as shown: We set the properties of this node as default.

*Screenshot showing the diagram of the nodes:*

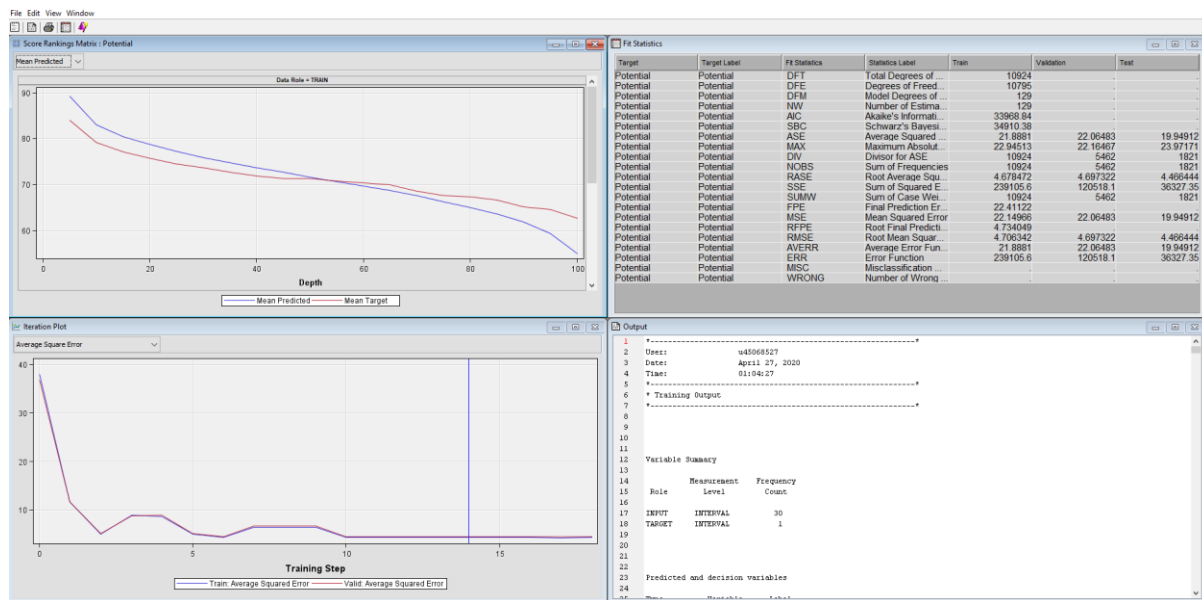


Right-click the **Model Comparison node** and click Run. In the Confirmation window, click Yes. In the Run Status window, click Results.

### Exploring the tree model results:

In our diagram workspace, when we right-click the AutoNeural node and click Results, the results are as shown:

## Screenshot showing the results:



## CONCLUSION

The Average Square error for this model is 21.881 which is very high when compared with all the previous models.

## 8. MODEL COMPARISON

Now that we have four candidate models (Regression, Decision Tree, Neural Network and AutoNeural) to use to predict the best target individuals for our mail solicitation, we can compare them to determine a champion model that we will use to score new data. We compare models and select a champion model that, according to an evaluation criterion explained below:

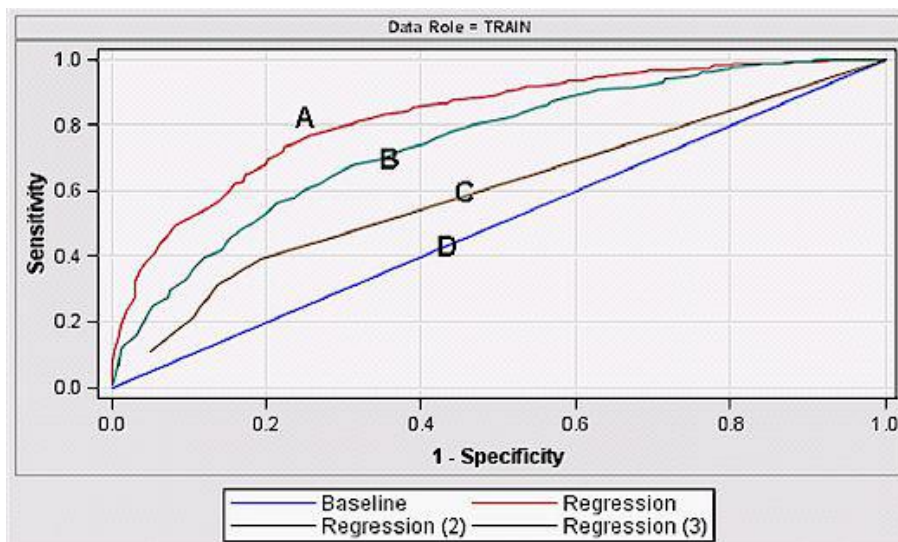
**Below are the explanations for each criterion:**

### AREA UNDER CURVE:

The SAS Enterprise Miner Model Comparison node computes the ROC chart. The ROC chart graphically displays Sensitivity versus 1-Specificity, or the true positive rate versus the false positive rate.

The true positive rate and the false positive rate are both measures that depend on the selected cutoff value of the posterior probability. Therefore, the ROC curve is calculated for all possible cutoff values.

Consider the following chart with four ROC curves, A, B, C, and D:



Each point on curves A, B, C, and D represents cutoff probabilities. Points that are closer to the upper right corner correspond to low cutoff probabilities. Points that are closer to the lower left corner correspond to high cutoff probabilities. The extreme points (1,1) and (0,0) represent rules where all cases are classified into either class 1 (event) or class 0 (non-event). For a given false positive rate (the probability of a non-event that was predicted as an event), the curve indicates the corresponding true positive rate, the probability for an event to be correctly predicted as an event.

Therefore, for a given false positive rate on the 1-Specificity axis, the true positive rate should be as high as possible. The different curves in the chart exhibit various degrees of concavity. The higher the degree of concavity, the better the model is expected to be. In the chart above, A appears to be the best model. Conversely, a poor model of random predictions, such as D, appears as a flat 45-degree line. Curves that push upward and to the left represent better models.

**AVERAGE SQUARED ERROR** — chooses the model with the smallest average squared error value.

**MISCLASSIFICATION RATE** — chooses the model with the lowest misclassification rate.

**ACCURACY:** The accuracy and misclassification are similar but inverse. The more the accuracy the lower will be the misclassification rate. Hence, with the accuracy chosen as a criterion choose the model with the highest accuracy.

Here, we have chosen the best model from each type of model we ran individually.

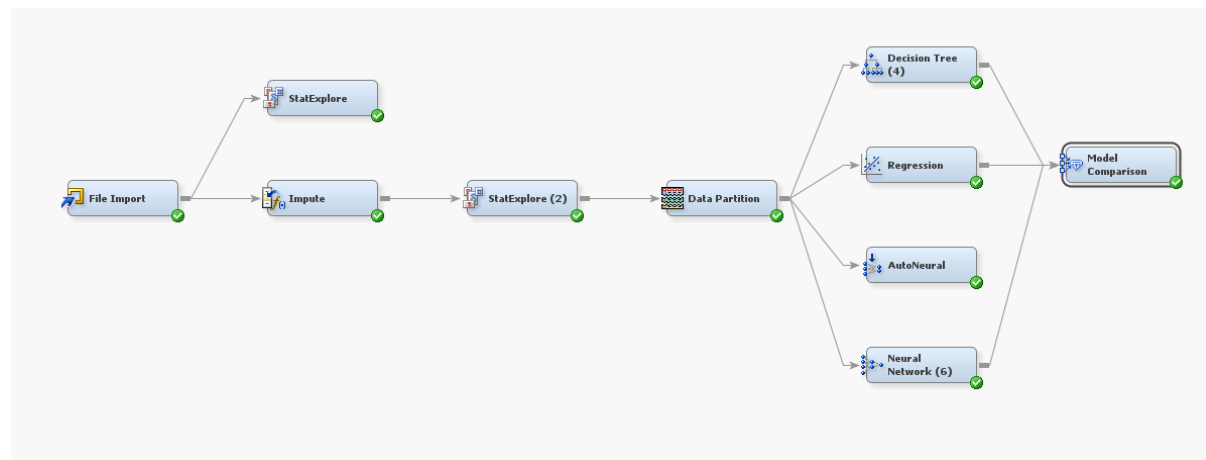
1. From the regression model the, node with the name Regression is selected
2. From the Decision Tree model, the node with the name Decision tree (4) is selected
3. From the Neural Network model, the node with the name Neural Network (6) is selected
4. From the AutoNeural model, the node with the name AutoNeural is selected

### Compare Models

We used the Model Comparison node to compare the models that we have built in this project and to select one as the champion model:

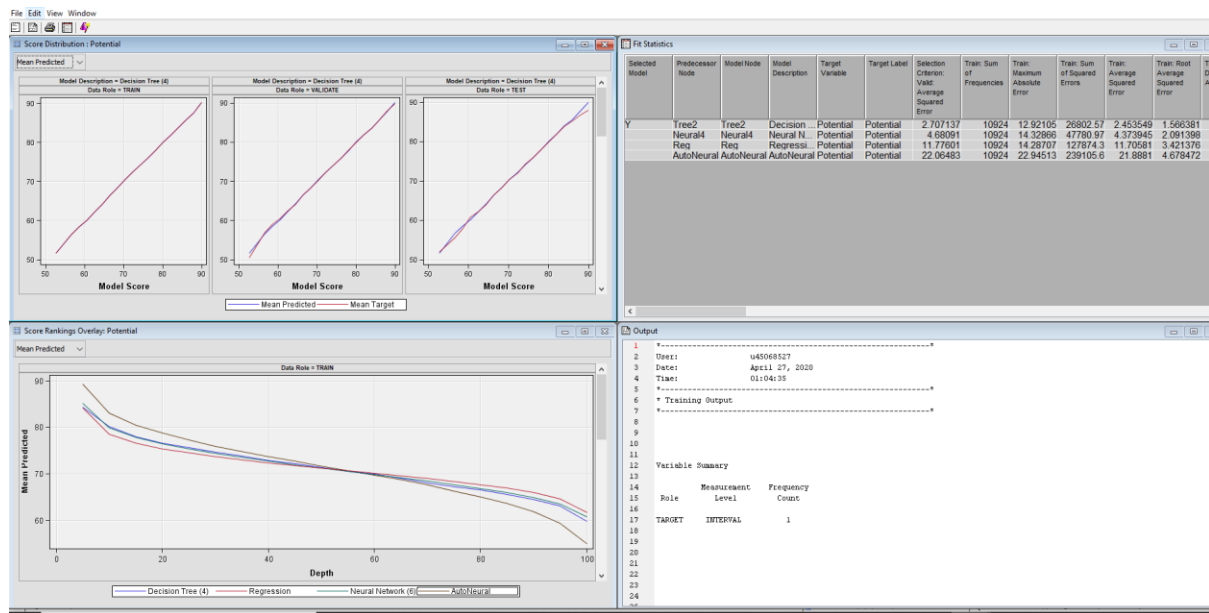
We selected the **Assess** tab on the Toolbar and selected the **Model Comparison** node icon. Dragged the node into the Diagram Workspace and connected all the four models to the **Model Comparison** node as shown:

*Screenshot showing the diagram of the nodes:*



In the Diagram Workspace, right clicked the **Model Comparison** node, and selected **Run** from the resulting menu. Clicked **Yes** in the Confirmation window that opens. In the window that appears when processing completes, clicked **Results**.

*Screenshot showing the results:*



*Screenshot showing fit statistics:*

Fit Statistics												
Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Valid Average Squared Error	Train: Sum of Frequencies	Train: Maximum Absolute Error	Train: Sum of Squared Errors	Train: Average Squared Error	Train: Root Average Squared Error	Train: Divis ASE
Y	Tree2	Tree2	Decision ...	Potential	Potential	2.707137	10924	12.92105	26802.57	2.453549	1.566381	
	Neural4	Neural4	Neural N...	Potential	Potential	4.68091	10924	14.32866	47780.97	4.373945	2.091398	
	Reg	Reg	Regressi...	Potential	Potential	11.77601	10924	14.28707	127874.3	11.70581	3.421376	
	AutoNeural	AutoNeural	AutoNeural	Potential	Potential	22.06483	10924	22.94513	239105.6	21.8881	4.678472	



Fit Statistics					
Model Selection based on Valid: Average Squared Error (_VASE_)					
Selected			Valid:	Train:	
Model	Model Node	Model Description	Average Squared Error	Average Squared Error	Train: Misclassification Rate
Y	Tree2	Decision Tree (4)	2.7071	2.4535	.
	Neural4	Neural Network (6)	4.6809	4.3739	.
	Reg	Regression	11.7760	11.7058	.
	AutoNeural	AutoNeural	22.0648	21.8881	.

In the Fit Statistics window, notice that the **Decision Tree (4)** model was selected as the champion model. The champion model has the value **Y** in the Selected Model column in the Fit Statistics window. In the model selection node, SAS Enterprise Miner selects the champion model based on the value of a single statistic. We can specify which statistic to use for selection in the node Properties Panel. Here, we chose ASE as selection criterion.

## 9. CONCLUSION

---

We have implemented a prediction system to predict if a player's performance is dependent on the age. It is evident that age of a player almost effects 50% of their performance. It is a key observation because it shows that potential has more than half dependency irrespective of player's attributes (fitness, position, current market value).

We have also created a best model to predict the player. For this we have selected the best one out of different regression models, neural networks, decision trees and a neural network and then compared all the models. Out of all the models Decision Tree stood out with best performance with least Average Squared error of 2.7071.c

These predictions will help us whether to consider a player's age in a critical situation and to predict the potential player which can help team to deliver a better performance.

## 10. REFERENCES AND TOOLS USED

---

### **References:**

Data Mining for Business Analytics: Concepts, Techniques, and Applications in Microsoft Excel with XLMiner [by Galit Shmueli, Nitin R. Patel, and Peter C. Bruce]

Getting started with SAS Enterprise Miner 14.2

Applied Linear Regression, Fourth Edition. Sanford Weisberg

Dataset from <https://www.kaggle.com/jinghuiwong/fifa-analysis-of-top-and-avg-players/data>

### **Tools Used:**

SAS Enterprise Miner