Machine Learning Approach to Predict the Invariant Mass of Dielectrons

Deepesh Narayan Raul

Department of Computer Science, S K Somaiya   
Mumbai, India

d.raul@somaiya.edu

***Abstract -* The elementary particles of the universe and its interactions is said to be concise by the Standard Model of high energy physics or particle physics [1]. Particle collider experiments has the capability to produce enormous and information-rich samples of data. Machine learning techniques can be used to develop how these data samples are interpreted, greatly expanding the discovery of potential present and future experiments. In this paper, data provided by Compact Muon Solenoid built on the Large Hadron Collider at CERN is analyzed to predict the invariant mass of two electrons using a statistical approach of machine learning.**

***Keywords: Machine Learning, CERN, Invariant Mass, Electrons, Regression, Feature Engineering.***

1. Introduction

Mid-1970s was when the Standard Model of particle physics was finalized upon the confirmation of quarks, since then, the evidence of top quark in 1995 [2], tau neutrino in 2000 [3-4] and the recent Higgs boson in 2012 [5] have solidified the credence of the Standard Model. This theory describes that every observable objects within this universe is made from basic blocks called elementary particles, ruled by the four forces.

The invariant mass also known as rest mass or intrinsic mass is the fraction of the total mass of an object or system of objects that is independent of the total motion of the system. In particle physics,

A collider experiment is used in particle physics research by colliding pair of particles at very high kinetic energy. The Conseil Europ ́een pour la Recherche Nucl ́eaire or as we call it, CERN houses the world’s largest and highest energy particle collider, the Large Hadron Collider (LHS) and the Compact Muon Solenoid (CMS) a particle physics detectors. The CMS can generate huge amount of data for particle collisions at 0.9 – 13 TeV. The invariant mass is calculated using a different equation in a particle collider experiments (if particles are highly relativistic, or massless),

Where is defined as the angular position of particle in terms of azimuthal angle and pseudo rapidity . is the observed transverse momentum.

Using the machine learning we can use robust computing system alongside modern algorithms to observe and analyze insights from enormous amount of data quickly and efficiently. In this paper, we are not only trying to demonstrate the statistical significance of machine learning models in the field of particle physics by predicting the invariant mass of dielectrons based on the observation from the CMS detectors but also manipulating the features of the dataset by creating new features such that these features show higher relationship with the target variable, Invariant mass and increase the performance of the ML models.

1. Dataset

The dataset used for this research is provided courtesy of the CERN open data portal [] contains observations of one hundred thousand dielectrons events in the invariant mass of 2-110 GeV captured by the Compact Muon Solenoid. This data is organized in a CSV spreadsheet file and include the following observations collected by the CMS:

* Run: The run number of the event.
* Event: Number of each event
* E1 and E2: Total energy of the 2 electrons in GeV
* px1, py1, pz1, px2, py2, and pz2: Components of the momentum of the electrons in GeV
* pt1 and pt2: Transverse momentum of the electrons in GeV
* phi1 and phi2: phi angle of the electrons 1 and 2 in rad
* eta1 and eta2: The 2 electrons pseudo-rapidity
* Q1 and Q2: The charge of the electrons
* Invariant Mass M: The invariant mass of the dielectrons in GeV

As this dataset comes directly from the CERN open portal, it is ensured that the observations are reliable, accurate and has been peer reviewed to be scientifically correct. The data released have been thoroughly analyzed and verified its accuracy through simulation events. Any results provided by this paper can be ensured to be true and accurate.

The dataset is further processed by examining, cleaning, and analyzing the data and its features. As duplicate data are an extreme case of nonrandom sampling, as well as they bias any of the models, leading to overfitting problems. In the case for the CERNs dataset, these duplicates are not real data nor is intentionally oversampled. Duplicate data are removed. After evaluations of the dataset, the target variable M i.e., the Invariant Mass has *NaN* values. The records are removed from the dataset instead of imputing the data as that may lead to false results.

1. Feature Engineering

This section provides the explanation of what manipulation of the features of the data was done and show the analysis of these new features. In statistics, correlation analysis is done to calculate the level of relation between one variable to another. In other words, it measures the linear association between 2 variables.

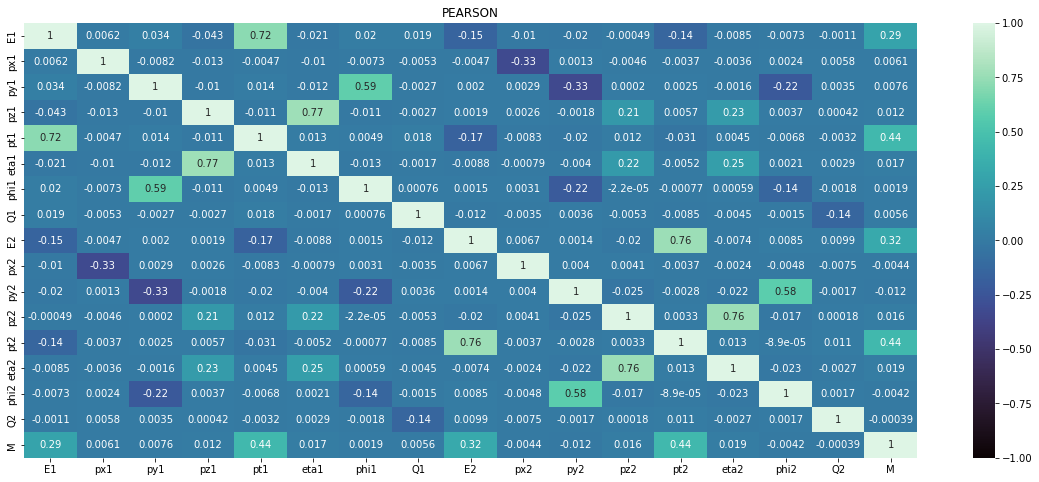


Figure 1 Heatmap for the correlation coefficients

Figure 1 is the heatmap for the Pearson’s correlation coefficient values of the dataset shows us the values between each feature. The major focus here is the coefficient value between the target value M and each independent variable. The heatmap shows that E1, E2, pt1 and pt2 features have high correlation values 0.29, 0.32, 0.44 and 0.44 respectively, with Invariant Mass compared to all the other features.

This research presents new features that are created using the original features.

* **E12:** the product of features E1 and E2.
* **pt12:** the product of features pt1 and pt2
* **Similarly, eta12, px12, py12, pz12, phi12,** the products for their respective features.

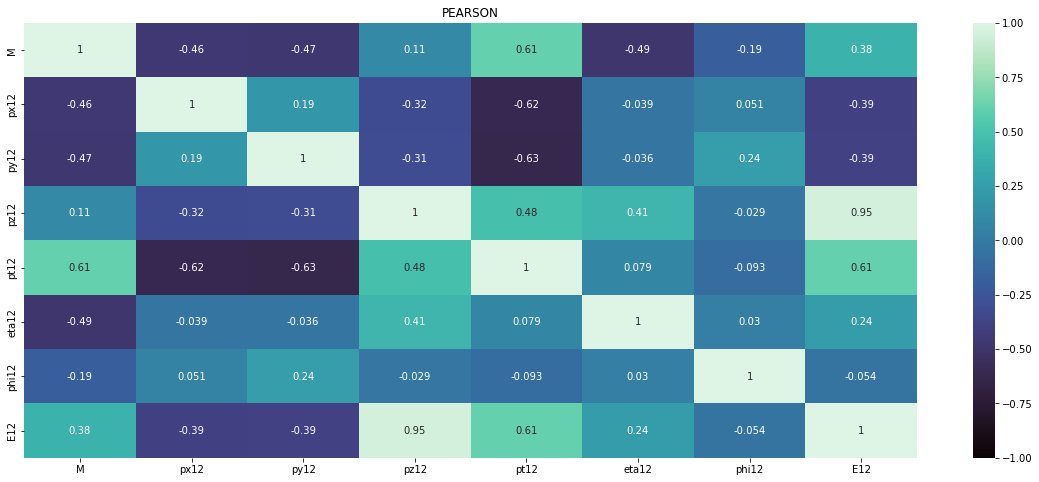


Figure 2 Correlation Heatmap of the new features

The heatmap from Figure 2 shows the correlation between the new features with the dependent variable M. With the min correlation value to be 0.11, we can say that that the new features have good relation to the dependent variable M

TABLE I  
Correlation coefficient value comparison

|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **Particle 1** | **Particle 2** | **Product** |
| px | 0.0061 | -0.0044 | -0.46 |
| py | 0.0076 | -0.012 | -0.47 |
| pz | 0.012 | 0.016 | 0.11 |
| pt | 0.44 | 0.44 | 0.61 |
| eta | 0.017 | 0.019 | -0.49 |
| phi | 0.0019 | -0.0042 | -0.19 |
| E | 0.29 | 0.32 | 0.38 |

The table I shows the comparison of the correlation coefficient values between the original two features and the new features created using them. We can infer that the new features highly correlated to M compared to the parent features.

With the introduction of new features to the dataset, 2 test needs to be done. First, perform the p-value test, to prove the significance of the correlation that is evaluated and secondly deal with the problem of multicollinearity.

The p-values test is a measure of significance to validate a hypothesis against the observed results or data. Here, the null hypothesis is that there is no correlation between the features and the target. The threshold for this test is , wherein the null hypothesis is rejected. The p-values for all the features, including the newly created ones, are calculated using Ordinary Least Square Regression model. Every feature has a p-value less than 0.05 and therefore null hypothesis is rejected, proving the correlations found are significant to use.

As new features are created, the problem of multicollinearity needs to be solved, where 2 or more independent variables are highly correlated with each other. High multicollinearity causes inaccurate results of regression analysis, due to the unstable and biased estimation of the regression coefficients, increases standard error and the variance of the coefficients which in turn also decreases the statistical power. Therefor the highly correlated independent variables are removed.

Multicollinearity is detected by calculating the variance inflation factor (VIF) of each feature in the dataset, it is the measure of how much the standard error of the estimate of the coefficient increased due to multicollinearity.

The threshold, accepted, for the VIF value is less than 10. Above which the feature is removed. For the original data all features (excluding M, the dependent variable) have VIF values below 10.

With the introduction of the new features to the dataset the VIF values are recalculated.

TABLE II  
Features with VIF value > 10

|  |  |
| --- | --- |
| **Features** | **VIF Value** |
| pt2 | 10.668826 |
| pz12 | 93.354042 |
| E12 | 118.840222 |

Table II reveals that 3 features have strong multicollinearity problems. As E12 and pt2 have high correlation with M compared to pz12, seen from figure 2, this feature is dropped and the new recalculated VIF values for all remaining features are within the threshold.

1. Model Selection

Ten regression models are selected and fitted to the original dataset and the new added feature dataset. Following are the regression models used for this research.

Decision Tree Regression is a regression model that uses the concept of decision tree. A model that uses a tree-like model of decisions to predict the target value. Gradient Boosting Regression is a machine learning ensemble technique which predicts target output by combining several weak learners in particular decision tree. LASSO stands for Least Absolute Shrinkage and Selection Operator is a regularization technique, specifically L1 regularization. Ridge regression is a regression model that uses L2 regularization technique. Random Forest Regressor takes same or multiple algorithms, and a model is put together that’s more effective than the original. PLS Regression also known as Partial Least Squares regression is a method that minimizes the predictors to a smaller set of uncorrelated components and performs least squares regression on these components. Elastic net linear regression is a regularized regression that overcomes the limitations of both the lasso and ridge techniques by linearly combining the L1 and L2 penalties of lasso and ridge models. Decision Tree regression, Gradient Boost, Lasso, Ridge, Elastic net, Random Forest Regressor, and the PLS regression are provided by the scikit-learn library.

XGBoost also known as eXtreme Gradient Boosting is a regularizing gradient boosting framework for many languages such as C++, python, scala etc. It is a open-sourced, distributed, scalable gradient-boosted decision tree (GBDT) library. Developed by The XGBoost Contributors. LightGBM, or Light Gradient Boosting Machine is framework of gradient boosting based on decision trees. It is developed by Microsoft and Guolin Ke. CatBoost is also a gradient boosting framework, developed by Yandex. This library is open-source and models can be built in various languages, including C++, python, R and many more. Unlike similar gradient boosting models, CatBoost grows oblivious trees, these trees are grown by enforcing the rule that all nodes at the same level, test the same predictor with the same condition and therefore bitwise operations can be used to calculate the index of the leaf.

Tables III and IV are the evaluation metrics of all the regression models that are fitted on the original dataset and the new dataset respectively. These metrics are calculated by taking the means of the R2 score and RMSE values evaluated across 10 – fold cross-validation. It is evident that the new features that were engineered have a positive impact on the performance across all the regression models. Keep in mind these models are fitted on their default hyperparameters. The four gradient boosting regression models, GBR, LGBM, XGBoost, and CatBoost have the best performance out of the other types of regression algorithms.

TABLE III  
Evaluation metrics using original dataset

|  |  |  |
| --- | --- | --- |
| **Model** | **R2 Score** | **RMSE** |
| DecisionTreeRegressor | 0.66747 | 14.55148 |
| CatBoostRegressor | 0.99037 | 2.47511 |
| ElasticNet | 0.39832 | 19.56982 |
| GradientBoostingRegressor | 0.74486 | 12.74279 |
| Lasso | 0.39835 | 19.56938 |
| LGBMRegressor | 0.95843 | 5.14299 |
| PLSRegression | 0.39183 | 19.67583 |
| RandomForestRegressor | 0.90155 | 7.91384 |
| Ridge | 0.39804 | 19.57434 |
| XGBRegressor | 0.96420 | 4.77334 |

TABLE IV  
Evaluation metrics using new dataset

|  |  |  |
| --- | --- | --- |
| **Model** | **R2 Score** | **RMSE** |
| DecisionTreeRegressor | 0.97820 | 3.72371 |
| CatBoostRegressor | 0.99699 | 1.38363 |
| ElasticNet | 0.81897 | 10.73765 |
| GradientBoostingRegressor | 0.98590 | 2.99343 |
| Lasso | 0.82125 | 10.66939 |
| LGBMRegressor | 0.99401 | 1.95138 |
| PLSRegression | 0.79007 | 11.56486 |
| RandomForestRegressor | 0.99092 | 2.40208 |
| Ridge | 0.82179 | 10.65290 |
| XGBRegressor | 0.99333 | 2.05787 |

The CatBoost regression is the best model with a R2 score of 0.99699 and a RMSE value of 1.38363. This paper will be using the CatBoost model for the prediction of invariant mass from this point onwards as it has the best performance out of all the models.

1. Hyperparameter tuning and normalization

In the previous section, it is mentioned that the models were evaluated on their default hyperparameters. The software framework Optuna [3] automates the steps involved in hyperparameter optimization. These optimization procedures seek to enhance the performance of a machine learning project while minimising the time and effort needed to execute it.

Bayesian optimization is a suitable option because it has been demonstrated to outperform other cutting-edge global optimization algorithms. [4] Only continuous hyper-parameters can be optimized using Bayesian methods; categorical ones cannot. TPE short for Tree-structured Parzen estimators uses trees to handle categorical hyper-parameters. [5]. A sampler using TPE algorithm provided by Optuna is used, to find the optimal hyperparameters for the CatBoost Model as shown in table V. This sampler is run over 100 trials and the final results has the best RMSE of 0.92046.

TABLE V  
Best Hyperparameters for CatBoost

|  |  |
| --- | --- |
| **Hyperparameters** | **Values** |
| subsample | 0.6571165219995333 |
| od\_wait | 46 |
| colsample\_bylevel | 0.6900521591366188 |
| random\_strength | 2 |
| l2\_leaf\_reg | 3.3108471723733297 |
| max\_depth | 9 |
| n\_estimators | 2215 |
| learning\_rate | 0.07942397557433238 |

References

1. W. N. Cottingham and D. A. Greenwood, *An Introduction to the Standard Model of Particle Physics,* 2nd Edition, Cambridge, 2007.
2. Abe, F.; et al. (CDF Collaboration) (1995). *"Observation of top quark production in collisions with the Collider-Detector at Fermilab".* Physical Review Letters. 74 (14): 2626–2631
3. Jackson, Judy; et al. *"Physicists find first direct evidence for tau neutrino at Fermilab"* (Press release). Batavia, IL: Fermilab. 2000
4. Kodama, K.; et al. (DONUT collaboration). *"Observation of tau neutrino interactions"*. Physics Letters B. 504 (3): 218–224, 2001.
5. Ellis, John; Gaillard, Mary K.; Nanopoulos, Dimitri V. (2012). *"A Historical Profile of the Higgs Boson"*. ArXiv, 2012,
6. Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. *Optuna: A Next-generation Hyperparameter Optimization Framework.* In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19). Association for Computing Machinery, New York, NY, USA, 2623–2631.
7. Snoek, Jasper, Larochelle, Hugo, and Adams, Ryan P. *Practical Bayesian optimization of machine learning algorithms*. In Advances in Neural Information Processing Systems, pp. 2951–2959, 2012.
8. James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. *Algorithms for hyper-parameter optimization.* In Proceedings of the 24th International Conference on Neural Information Processing Systems (NIPS'11). Curran Associates Inc., Red Hook, NY, USA, 2546–2554.