ECSE 551 Group 5 Mini-Project 1

Mahta Amini
McGill University
mahta.amini@mail.mcgill.ca

Sara Yabesi
McGill University
sara.yabesi@mail.mcgill.ca

Baharan Nouriinanloo
McGill University
baharan.nouriinanloo@mail.mcgill.ca

Abstract

In this project, logistic regression was utilized to address two binary classification problems: the white wine quality and the blood in kidney datasets. The study commenced by developing simple models to establish baseline results, which were then improved by conducting various experiments. To arrive at the best set of features, we used feature importance to build our models with the best set of features. Removing the least important feature (Heart rate) in the kidney disease dataset yielded better results. All features were needed in the white wine datasets to build the best model. Some other techniques that enhanced the accuracy of models are log transformation, second and third order of features. Overall, the logistic regression model findings indicate that these modifications can improve the accuracy and robustness of the model for both datasets.

1 Introduction

Logistic regression is a powerful machine-learning tool for performing binary classification tasks. The algorithm uses an iterative optimization algorithm, such as gradient descent, to estimate the coefficients or weights of the model. Using the logistic function to convert the estimated log-odds ratio of a given class, which is derived as a linear combination of features, we can predict the probability of a particular outcome based on input data [1]. This report applies logistic regression to two classification tasks: predicting the presence of blood in a patient's kidney based on their medical history and assessing the quality of white wine based on certain characteristics.

To create the most effective models for these tasks, the datasets are analyzed, feature engineering is performed, and the optimization algorithm parameters are carefully considered. It is crucial to understand the data set to build an accurate machine learning model [2]. Section 2 describes the datasets in detail. It describes how we used the feature importance technique [3] to select the best set of features. This technique is used to evaluate the relationship between features and class labels. We also demonstrate the distribution of features in both datasets and applied log-transformation if needed. To test the performance of the model with higher complexity, we also add different order of parameters to the model.

The effectiveness of different feature engineering techniques is evaluated in Section 3, with analysis of the influence of several optimization parameters on model performance, including tolerance, learning rate, and initial weights. A 10-fold cross-validation approach is employed to determine the most effective models, based on the highest accuracy obtained [4]. Furthermore, the number of iterations required for the optimization algorithm to converge is determined by examining the point at which additional iterations no longer result in significant decreases in the cost function [5].. Finally, in Section 4, the report summarizes the key findings and suggests areas for future research. Overall, this report demonstrates the importance of understanding the data and carefully optimizing machine learning models to achieve the best possible results in binary classification tasks.

After training the models, we found that the best learning rate in both datasets is $\frac{1}{k+1}$ where k is the iteration number. The best ϵ for the kidney disease dataset is 10^{-6}, while for the white wine quality dataset, the best $\epsilon = 10^{-5}$. Removing the least important feature (Heart rate) in the kidney disease dataset yielded better results. All features were needed in the white wine datasets to build the best model. In the white wine dataset, the log of Total phenols and Proanthocyanins has the best accuracy. In the kidney disease dataset, the log of Insulin and Age has the best accuracy.

2 Datasets

Before applying logistic regression to a dataset, it is crucial to analyze the data thoroughly.

2.1 White Wine Quality

This data set contains 1599 samples and ten features. These features include Alcohol, Malic acid, Ash, Alkalinity of ash, Magnesium, Total phenols, Flavanoids, Nonflavanoid phenols, Proanthocyanins, and Hue. We double-check to see that all features have been standardized to provide normalized values. The class label of the white wine data set is a crucial element in determining the quality of wine, with a score of 1 being assigned to wines of high quality and 0 assigned to wines of low quality. By examining the dataset, we found that it consists of 855 samples of high-quality wines and 744 samples of low-quality wines. This distribution is plotted in Figure 1.

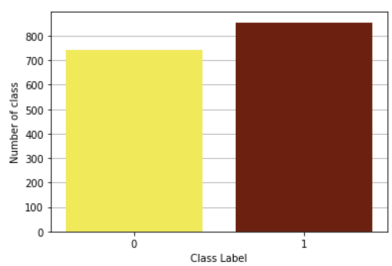


Figure 1: Distribution of class labels in white wine dataset

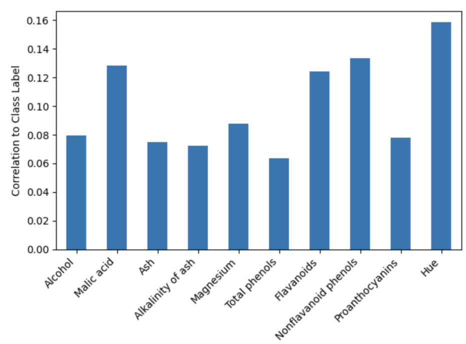


Figure 2: Importance of each feature to the class label in the white wine dataset

No outliers existed in the dataset. It also did not have any null values. By checking for duplicate rows, we found 244 duplicate row in white wine dataset. We depicted the correlation of features to the class label in Figure 2 using Feature Importance. Most features are almost dependent on the class label. However, Total phenols and Alkalinity of ash have the least correlation to the class label. For this reason, removing these independent features from the logistic regression is more reasonable. This process is explained in more detail in Section 3.5.

We demonstrated the distributions of the features of white wine dataset in Figure 3. It would be desirable if all the features were normally distributed and had similar scales. However, some features such as Ash, Alkalinity of ash and Total phenols have skewed distributions. To enhance the performance of the model, a log-transformation that would normalize these features could be employed [6]. This process in explained in Section 3.4.

There exist multiple techniques for determining the optimal approach to selecting features to include, exclude, scale or multiply to produce the most effective model. Other than Feature Importance technique, Mutual Information [7] or Correlation-based Feature Selection [8] can also be named. We demonstrated the correlation between features in Figure 4. It would be cautious about identifying which features have a high degree of correlation to each other. Keeping just one of them may be sufficient in cases with a significant correlation between features. Utilizing all correlated features could result in overfitting, which would ultimately reduce the accuracy[9].Some features such as Ash and Alcohol are highly correlated. We consider removing highly correlated features and training different models with them.

2.2 Blood in Kidney

The blood in kidney dataset contains medical information, with 330 inputs and nine unique features used as predictor variables. These features include Pregnancies (number of times pregnant), Glucose

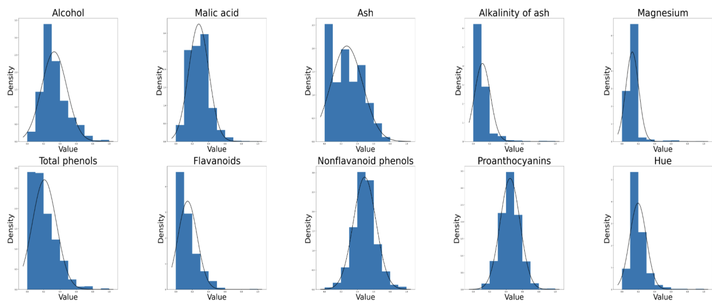


Figure 3: Distribution of white wine features

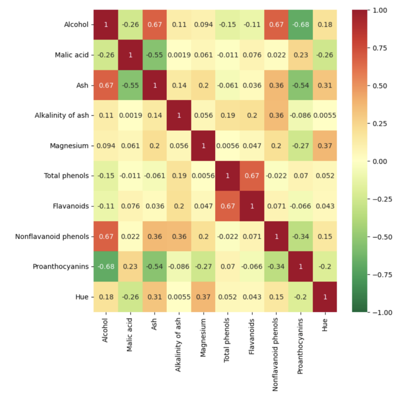


Figure 4: Correlation between white wine features

(plasma glucose concentration after a 2-hour oral glucose tolerance test), Blood Pressure (diastolic blood pressure in mm Hg), Heart Rate, Skin Thickness (triceps skin fold thickness in mm), Insulin (2-hour serum insulin in mu U/ml), BMI (body mass index), Diabetes Pedigree Function, and Age (in years).

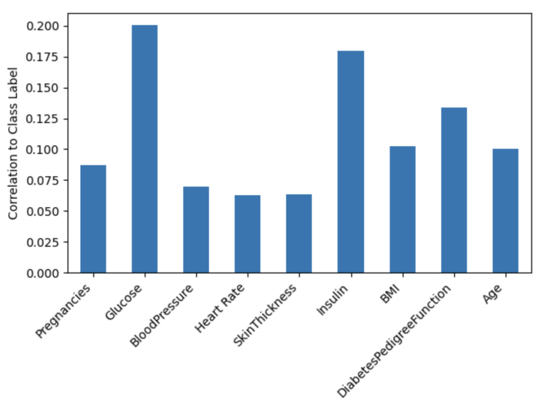


Figure 5: Importance of each feature to the class label in the blood in kidney dataset

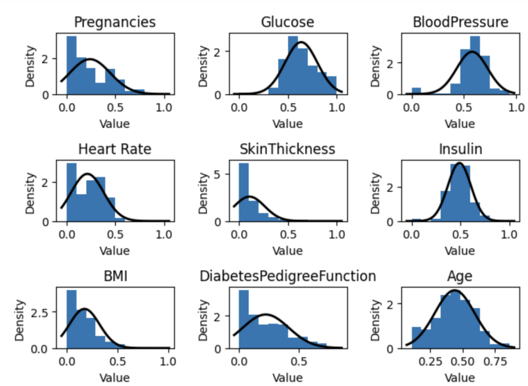


Figure 6: Distribution of blood in kidney features

In blood in kidney dataset, out of the 330 inputs, 165 have a class label of 1, indicating the presence of blood, while the other 165 have a class label of 0, indicating the absence of blood. No outliers existed in the dataset. It also did not have any null values. The kidney disease dataset also did not have any duplicate rows.

We have taken a page from our analysis of the white wine quality dataset and looked at the correlation between the features and class labels, which is plotted in Figure 5. As discussed earlier, we are exploring the possibility of removing independent features to improve the model's accuracy. We understand that some features like Heartrate, SkinThickness and BloodPressure have the least correlation to class labels.

In Figure 6, we have also plotted the distribution of the features in the dataset. Interestingly, some of the features, like SkinThickness and BMI, do not have a normalized distribution. This is taken into consideration as our logistic regression classification model is being fine-tuned.

In both datasets, we consider building models with higher complexity of features and training those models. Different combinations and orders of features are tested in section 3.6

3 Results

The proposed logistic regression model is built based on five functions. The fit function is utilized to determine the weights by performing gradient descent when working with a training dataset. A

predict function to anticipate outcomes using a validation dataset. The sigmoid function takes any real value and maps it between 0 and 1. the accu_eval function compares the anticipated results with the actual data. Finally, the calculate loss function calculates the model's cross-entropy loss.
The logistic regression's performance is evaluated using a standard 10-fold cross-validation method. During each training, multiple values are stored. The learning rate, the epsilon, the number of iterations required for convergence, the elapsed time for the training, the model weights and whether or not the maximum number of iterations was reached during training. The model's accuracy is also calculated as the mean accuracy of the folds in k-fold validation.

3.1 Shuffling the data

It was found that both the white wine dataset and the kidney disease dataset are ordered by the class label. If the data is not shuffled, it would lead to validation/training sets with only one class of data. Therefore, it is necessary to shuffle the rows of the dataset once it is read. This not only prevents overfitting and reduces variance but also helps to generalize the algorithm.

3.2 Learning rate

To find out the best learning rates, different variables are tested. These include independent variables (constant numbers like $1/8$) and variables dependent on other parameters. For example, k is the iteration number inside the while loop in our fit function, and n is the number of samples. The performance of all these learning rates on both datasets is presented in table 1. The model failed to converge in a reasonable amount of iterations for constant learning rates for the white wine dataset. The only constant learning rate with acceptable performance in the kidney disease dataset is 0.01. However, the best learning rate for both datasets is $\frac{1}{1+k}$. While having the highest accuracy, it still holds an acceptable iteration count. If the learning rate is too small the model will take a long time to converge, and the training may get stuck in a suboptimal solution, never reaching the global minimum. On the other hand, if the learning rate is too large, the model may oscillate forever.

Learning rate	Learning rate type	White wine		Kidney disease	
		Iteration count	Accuracy%	Iteration count	Accuracy%
0.01	Constant	150000	failed	956	0.7495
1/2	Constant	150000	failed	150000	0.6832
1/4	Constant	150000	failed	150000	0.6835
1/8	Constant	150000	failed	150000	0.6832
0.05	Constant	150000	failed	150000	0.6885
1/k	dependent	4032	0.7350	2220	0.7510
1/(k+1)	dependent	2753	0.7381	1476	0.7520
1/n	dependent	1362	0.7292	2048	0.7495
1/10n	dependent	74	0.5366	3516	0.7414
1/100n	dependent	2	0.5351	2	0.5168

Table 1: The performance of different learning rates on both datasets.

3.3 Stopping Condition

The weights for this project are trained until the difference between the current weight vector and the previous weight vector, denoted as $||w_k - w_{k-1}||$, is less than the tolerance value ϵ . In this section, different values for ϵ are experimented with, ranging from 10^{-2} to 10^{-9}. The training result is demonstrated in Table 2. Due to the page limitations, the table is summarized. But, the full model configurations for each dataset are available inside the code folder. For the white wine dataset, there is only a marginal improvement in accuracy in 10^{-6} compared to 10^{-5}, which comes at the expense of a significant increase in the number of iterations. Therefore, is it more reasonable to choose 10^{-5}. For the kidney disease dataset, $\epsilon = 10^{-6}$ is chosen since it increases the accuracy by just a small cost in time and iteration number.

3.4 Log Transformation

As discussed in Section 2, some features have skewed distributions, and standardization can help improve their distribution. To address this issue, one solution is to transform these features into distributions that resemble a Gaussian distribution by taking the logarithm of their values [10]. An experiment was conducted to examine the impact of a log transformation on the accuracy of the

Epsilon	White wine			Kidney disease		
	Iteration count	Time	Accuracy%	Iteration count	Time	Accuracy%
1e-2	235	0.138	0.7298	84	0.009	0.7370
1e-5	2753	1.104	0.7381	636	0.027	0.7495
1e-6	5496	2.174	0.7382	876	0.072	0.7515
1e-9	36990	16.286	0.7381	22164	22164	0.7508

Table 2: The performance of different epsilons on both datasets.

logistic regression model. In the white wine dataset, the log of Total phenols and Proanthocyanins have the best accuracy. In the kidney disease dataset, the log of Insulin and Age have the best accuracy.

3.5 Removing Features

We performed Feature Importance in Section 2. The results provided evidence that certain features are statistically independent to some extent from the class label. In the white wine dataset, the features that are most likely to be independent of the class label are Total phenols, Alkalinity of ash, and Ash. For the Kidney disease data set, Heart Rate and SkinThickness are most likely to be independent of the class label.
The experiment involved removing each of the potential independent features individually, as well combinations of them at once, and recording any resulting increase in accuracy. The kidney disease dataset exhibited an accuracy improvement of 1.2% when the Heartrate feature was removed. However, in the white wine dataset, the model's accuracy was not enhanced by removing any of the features.

3.6 Higher Complexity Model

When building models, it is sometimes beneficial to add the exponentiation of features to the model to improve its accuracy. This can create new dimensions and capture non-linear relationships between variables. In this project, we have performed exponentiation of features to increase the model's complexity. We have used different combination features in the datasets and recorded the model's accuracy. The best model in the white wine dataset includes the order two of alkalinity of ash and magnesium features. The most accurate model in the kidney disease dataset has the order three of the SkinThickness feature.

4 Discussion and Conclusions

In this mini-project, logistic regression was applied to two datasets with the aim of calculating the accuracy of 10-fold cross-validation. Several methods were used to improve the accuracy, including selecting important features, performing log transformation on features with non-normal distributions, and exponentiation. In conclusion, the application of these methods to logistic regression models can be useful in enhancing their performance and accuracy. However, it is important to note that the choice of method should depend on the characteristics of the dataset and the objectives of the study.

4.1 Future Investigation

As a future investigation, we could explore the use of other machine learning algorithms such as decision trees, random forests, support vector machines, or neural networks and measure their accuracy on the datasets. In addition, it is practical to study the impact of a feature selection method, like Recursive Feature Elimination on the model's accuracy. This method measures the correlation of features to the class label [11].

Another step for future investigation would be to collect external data on the similar subject and test the model's accuracy on those new data. The external dataset may include different types of noise or errors that the model has not encountered before in the cross-validation procedure. By doing so, we would be able to evaluate the generalizability of the model and investigate its accuracy.

5 Contribution Statement

Sara Yabesi: preparing the data, implementing the algorithm, evaluating the model, and preparing the report.

Mahta Amini: preparing the data, implementing the algorithm, evaluating the model, and preparing the report.
Baharan Nouriinanloo: preparing the data, implementing the algorithm, evaluating the model, and preparing the report.

References

[1] Muge Mutis, Ufuk Beyaztas, Gulhayat Golbasi Simsek, and Han Lin Shang. A robust scalar-on-function logistic regression for classification. *Communications in Statistics-Theory and Methods*, pages 1–17, 2022.

[2] Ivan Reyes-Amezcua, Daniel Flores-Araiza, Gilberto Ochoa-Ruiz, Andres Mendez-Vazquez, and Eduardo Rodriguez-Tello. Macfe: A meta-learning and causality based feature engineering framework. In *Advances in Computational Intelligence: 21st Mexican International Conference on Artificial Intelligence, MICAI 2022, Monterrey, Mexico, October 24–29, 2022, Proceedings, Part I*, pages 52–65. Springer, 2022.

[3] Alexander Zien, Nicole Krämer, Sören Sonnenburg, and Gunnar Rätsch. The feature importance ranking measure. *arXiv preprint arXiv:0906.4258*, 2009.

[4] Daniel S Soper. Hyperparameter optimization using successive halving with greedy cross validation. *Algorithms*, 16(1):17, 2022.

[5] Chao-Ying Joanne Peng, Kuk Lida Lee, and Gary M Ingersoll. An introduction to logistic regression analysis and reporting. *The journal of educational research*, 96(1):3–14, 2002.

[6] Husam AH Al-Najjar, Biswajeet Pradhan, Bahareh Kalantar, Maher Ibrahim Sameen, M Santosh, and Abdullah Alamri. Landslide susceptibility modeling: An integrated novel method based on machine learning feature transformation. *Remote Sensing*, 13(16):3281, 2021.

[7] Pablo A Estévez, Michel Tesmer, Claudio A Perez, and Jacek M Zurada. Normalized mutual information feature selection. *IEEE Transactions on neural networks*, 20(2):189–201, 2009.

[8] Mark A Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999.

[9] Pooja Rani, Rajneesh Kumar, and Anurag Jain. A hybrid approach for feature selection based on correlation feature selection and genetic algorithm. *International Journal of Software Innovation (IJSI)*, 10(1):1–17, 2022.

[10] FENG Changyong, WANG Hongyue, LU Naiji, CHEN Tian, HE Hua, LU Ying, et al. Log-transformation and its implications for data analysis. *Shanghai archives of psychiatry*, 26(2):105, 2014.

[11] Puneet Garg and Shailendra Narayan Singh. Analysis of ensemble learning models for identifying spam over social networks using recursive feature elimination. In *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pages 713–718. IEEE, 2021.

6 Appendix

Kidney_disease_analysis

February 19, 2023

[27]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm
```

[28]:
```
from google.colab import drive

drive.mount('/content/gdrive/', force_remount=True)
kd_data = pd.read_csv('/content/gdrive/MyDrive/kidney_disease.csv')
print(kd_data.shape)
```

```
Mounted at /content/gdrive/
(329, 10)
```

[29]:
```
kd_data.columns = ['Pregnancies', 'Glucose', 'BloodPressure', 'Heart Rate',␣
 ↪'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction',␣
 ↪'Age','class_label']
```

[30]:
```
kd_feature_names = kd_data.columns[:-1]

kd_feature_number = len(kd_feature_names)

kd_features = kd_data.iloc[:,0:-1]
kd_data_cl_number = len(kd_data)
```

[31]:
```
kd_data.describe()
```

[31]:
```
       Pregnancies     Glucose  BloodPressure  Heart Rate  SkinThickness  \
count   329.000000  329.000000     329.000000  329.000000     329.000000
mean      0.240837    0.635717       0.579127    0.214301       0.106602
std       0.207416    0.165684       0.148497    0.167121       0.156054
min       0.000000    0.000000       0.000000    0.000000       0.000000
25%       0.058824    0.512560       0.524590    0.000000       0.000000
50%       0.176470    0.613070       0.590160    0.232320       0.057920
75%       0.411760    0.738690       0.655740    0.333330       0.153660
max       1.000000    1.000000       0.934430    1.000000       1.000000
```

```
          Insulin         BMI  DiabetesPedigreeFunction         Age  \
count  329.000000  329.000000                329.000000  329.000000
mean     0.489131    0.177812                  0.223759    0.440215
std      0.118597    0.148117                  0.192474    0.153893
min      0.000000    0.000000                  0.000000    0.107690
25%      0.420270    0.075149                  0.066667    0.338460
50%      0.490310    0.132790                  0.183330    0.430770
75%      0.552910    0.246370                  0.350000    0.553850
max      1.000000    0.961140                  0.816670    0.861540

       class_label
count   329.000000
mean      0.498480
std       0.500759
min       0.000000
25%       0.000000
50%       0.000000
75%       1.000000
max       1.000000
```

[32]:
```
kd_data.duplicated().sum()
print("is duplicate:",kd_data.duplicated().sum())

print("is null:",kd_data.isnull().values.any())

print("maxmium values",kd_data.max(axis=0))
print("minimum values",kd_data.min(axis=0))
```

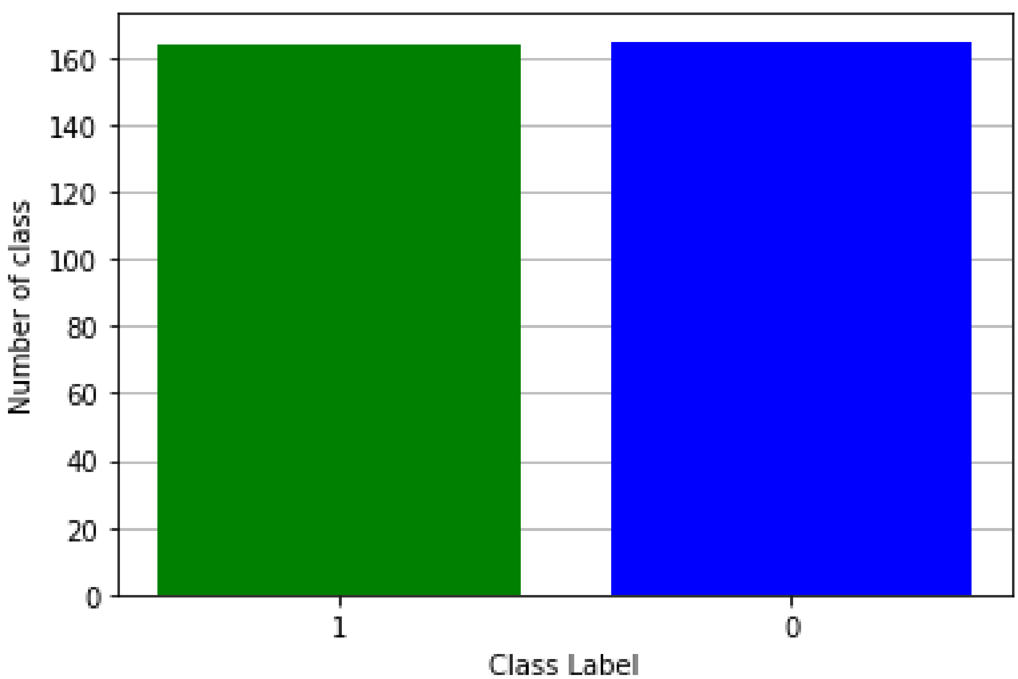
```
is duplicate: 0
is null: False
maxmium values Pregnancies                 1.00000
Glucose                     1.00000
BloodPressure               0.93443
Heart Rate                  1.00000
SkinThickness               1.00000
Insulin                     1.00000
BMI                         0.96114
DiabetesPedigreeFunction    0.81667
Age                         0.86154
class_label                 1.00000
dtype: float64
minimum values Pregnancies                 0.00000
Glucose                     0.00000
BloodPressure               0.00000
Heart Rate                  0.00000
SkinThickness               0.00000
Insulin                     0.00000
```

```
BMI                         0.00000
DiabetesPedigreeFunction    0.00000
Age                         0.10769
class_label                 0.00000
dtype: float64
```

[33]:
```
# distribution of class 0 and class 1
kd_label_counts = kd_data["class_label"].value_counts()
print(kd_label_counts)
unique_labels = kd_data["class_label"].unique()

fig, ax = plt.subplots()
ax.grid(zorder=1, axis="y")
ax.bar(unique_labels, kd_label_counts, zorder=2,color=['blue', 'green'])
ax.set_xticks([0,1])
ax.set_xticklabels(unique_labels)
ax.set_ylabel("Number of class")
ax.set_xlabel("Class Label")
```

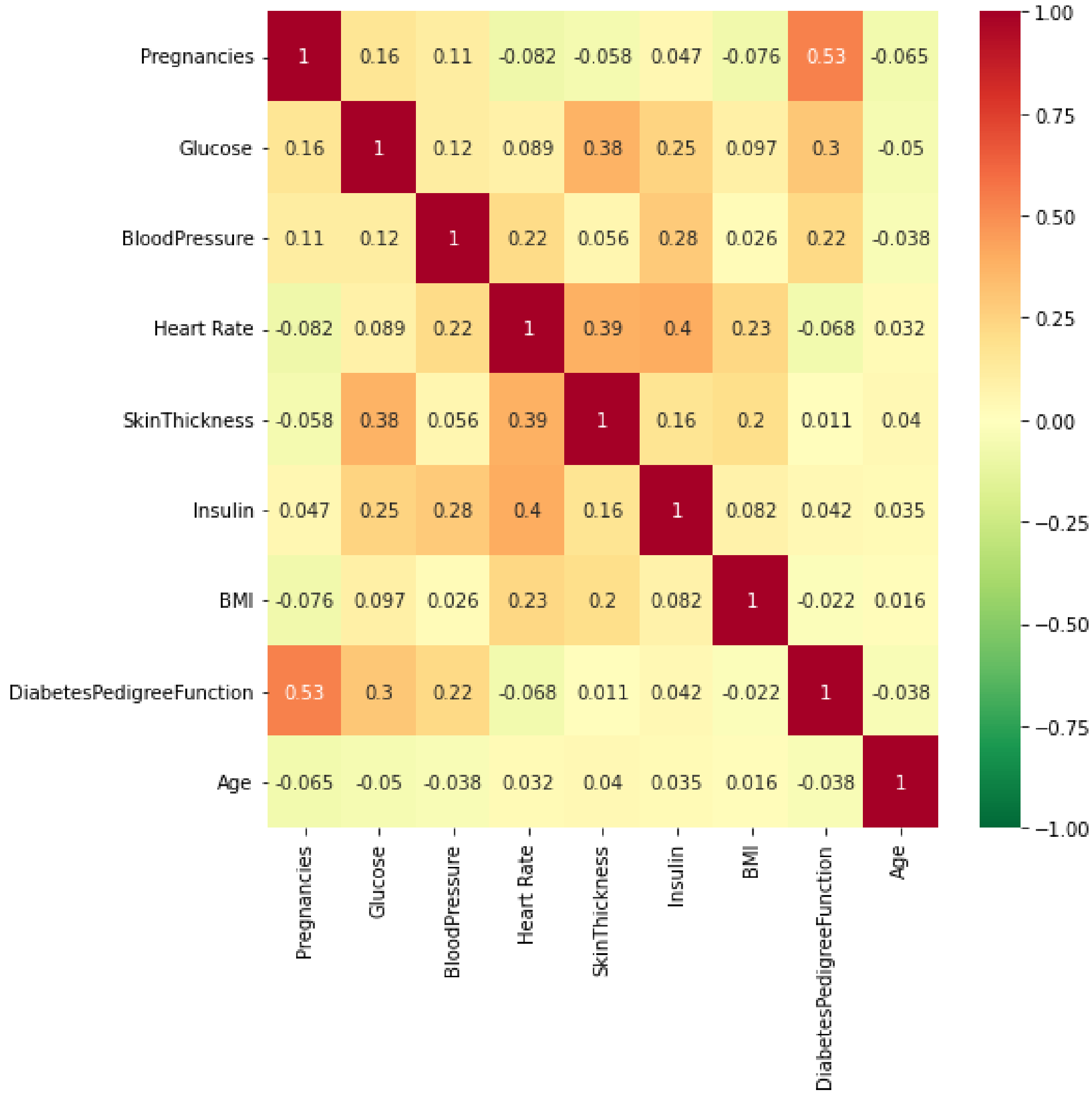
```
0    165
1    164
Name: class_label, dtype: int64
```

[33]:
```
Text(0.5, 0, 'Class Label')
```



[34]:

```
#draw heatmap of kidney disease
kd_corr = kd_features.corr()
plt.figure(figsize=(8,8))
sns.heatmap(kd_corr, cmap='RdYlGn_r', annot=True, vmin=-1, vmax=1)
plt.show()
```

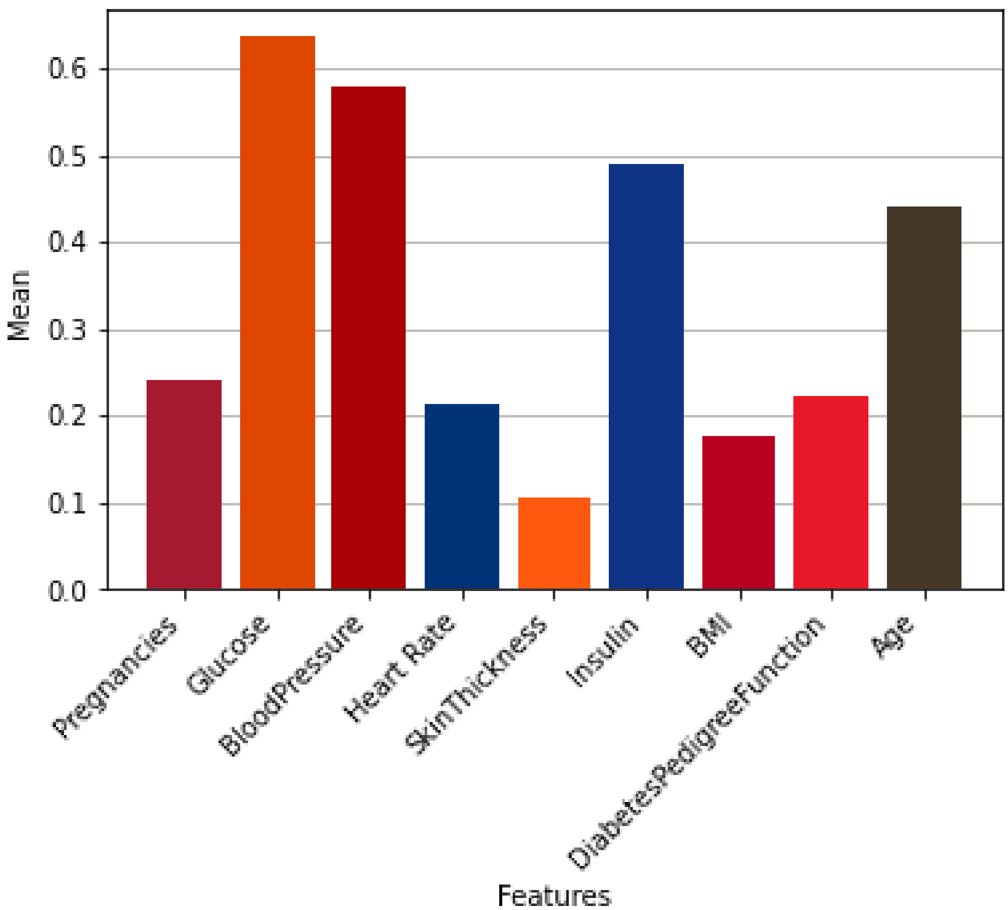


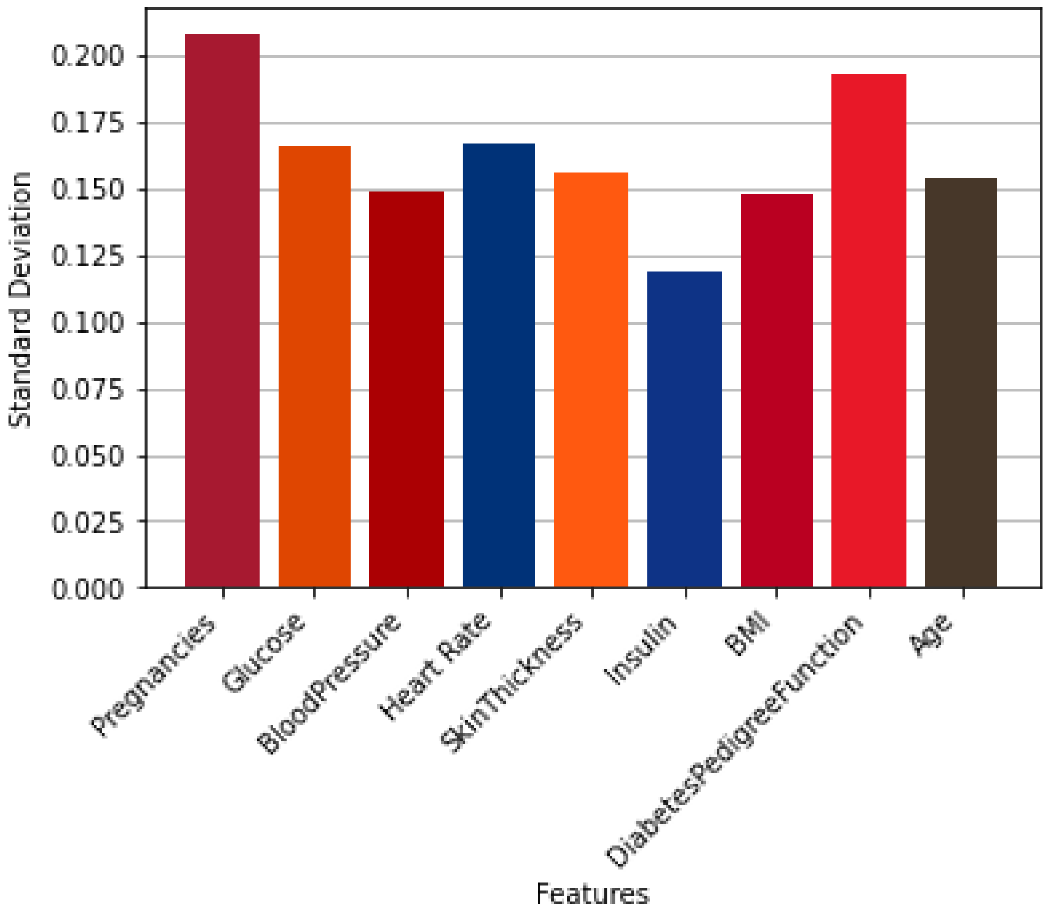
[35]:

```
# Compute mean of each feature
means = kd_features.mean()

# Compute standard deviation of each feature
std = kd_features.std()

# Plot mean and standard deviation of each attribute
```

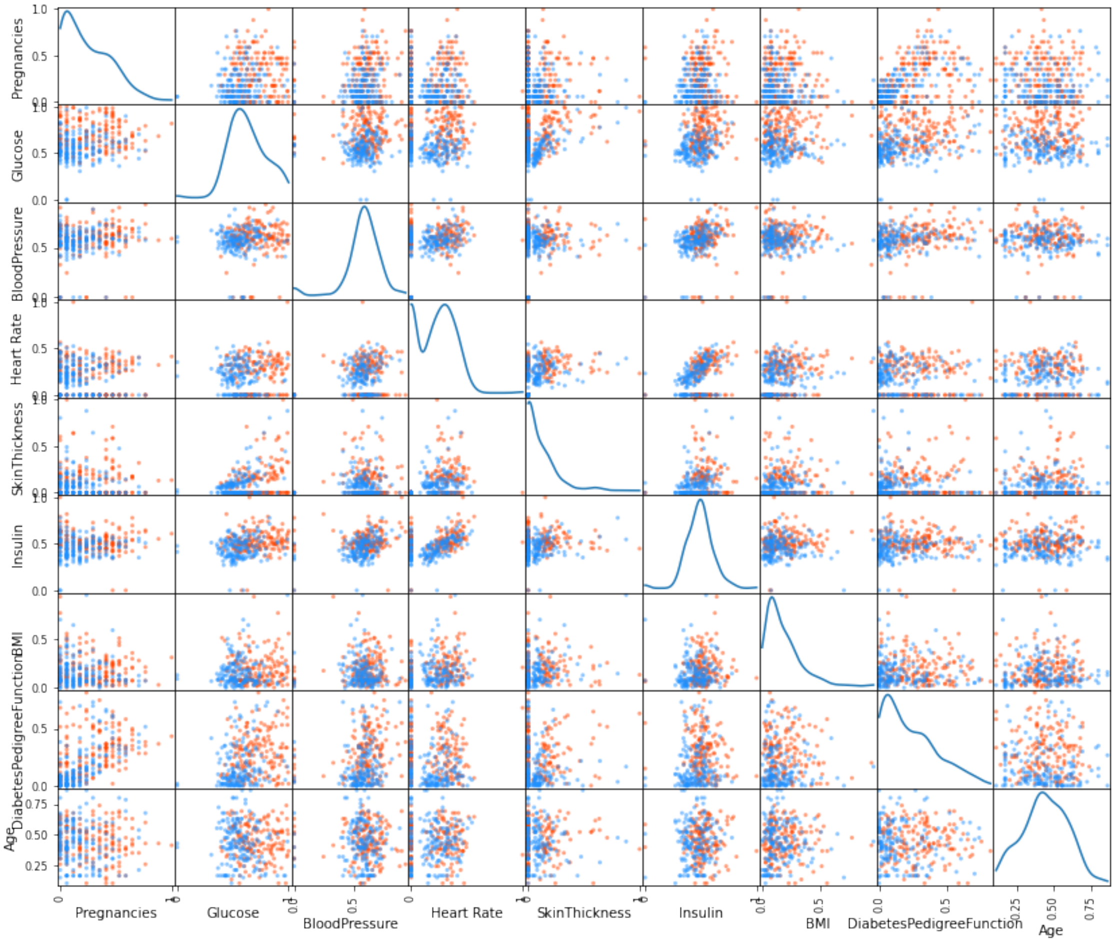
```
kd_colors=['#A71930', '#DF4601', '#AB0003', '#003278', '#FF5910', '#0E3386',␣
↪'#BA0021', '#E81828', '#473729']
#, '#D31145', '#0C2340', '#005A9C', '#BD3039', '#EB6E1F', '#C41E3A', '#33006F',␣
↪'#C6011F', '#004687', '#CE1141', '#134A8E', '#27251F', '#FDB827', '#0C2340',␣
↪'#FD5A1E', '#00A3E0', '#ffc52f', '#003831', '#005C5C', '#E31937', '#8FBCE6']


fig, ax = plt.subplots()

ax.grid(zorder=1, axis="y")
ax.bar(kd_feature_names, means, zorder=2,color=kd_colors)
plt.xticks(rotation=45,ha='right')
ax.set_xlabel("Features")
ax.set_ylabel("Mean")


fig, ax = plt.subplots()
ax.grid(zorder=1, axis="y")
ax.bar(kd_feature_names, std, zorder=2,color=kd_colors)
ax.set_xlabel("Features")
plt.xticks(rotation=45,ha='right')
ax.set_ylabel("Standard Deviation")
```

[35]: Text(0, 0.5, 'Standard Deviation')


0.6
0.5
0.4
0.3
0.2
0.1
0.0
Mean
Pregnancies
Glucose
BloodPressure
Heart Rate
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age
Features



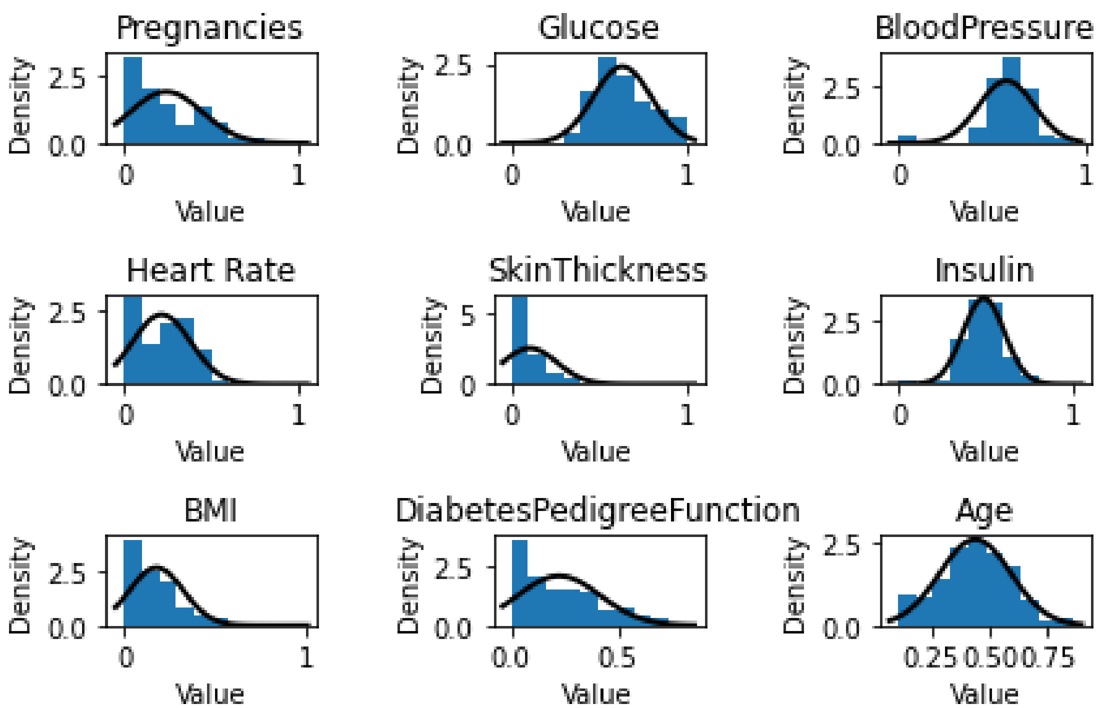
[36]:
```
color = ['dodgerblue', 'orangered']
colors = kd_data['class_label'].map(lambda x: color[x])
pd.plotting.scatter_matrix(kd_features, figsize= (14, 12), diagonal='kde',␣
    ↪color=colors);
```



[37]:
```python
#distribution of features
def draw_distributon_graph(x_features,names):
  kd_normal_fig, axis = plt.subplots(3, 3)
  col = 0

  for x, ax in enumerate(axis.ravel()):
      ax.hist(x_features.loc[:, names[col]], density=True)
      mu, std = norm.fit(x_features.loc[:, names[col]])
      xmin, xmax = ax.get_xlim()
      x = np.linspace(xmin, xmax, 100)
      p = norm.pdf(x, mu, std)
      ax.plot(x, p, 'k', linewidth=2)
      ax.set_title(names[col])
      ax.set_xlabel("Value")
      ax.set_ylabel("Density")
      col += 1
  kd_normal_fig.tight_layout()
```

```
print(kd_feature_names)
draw_distributon_graph(kd_features,kd_feature_names)
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'Heart Rate',
       'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age'],
      dtype='object')
```



[38]:

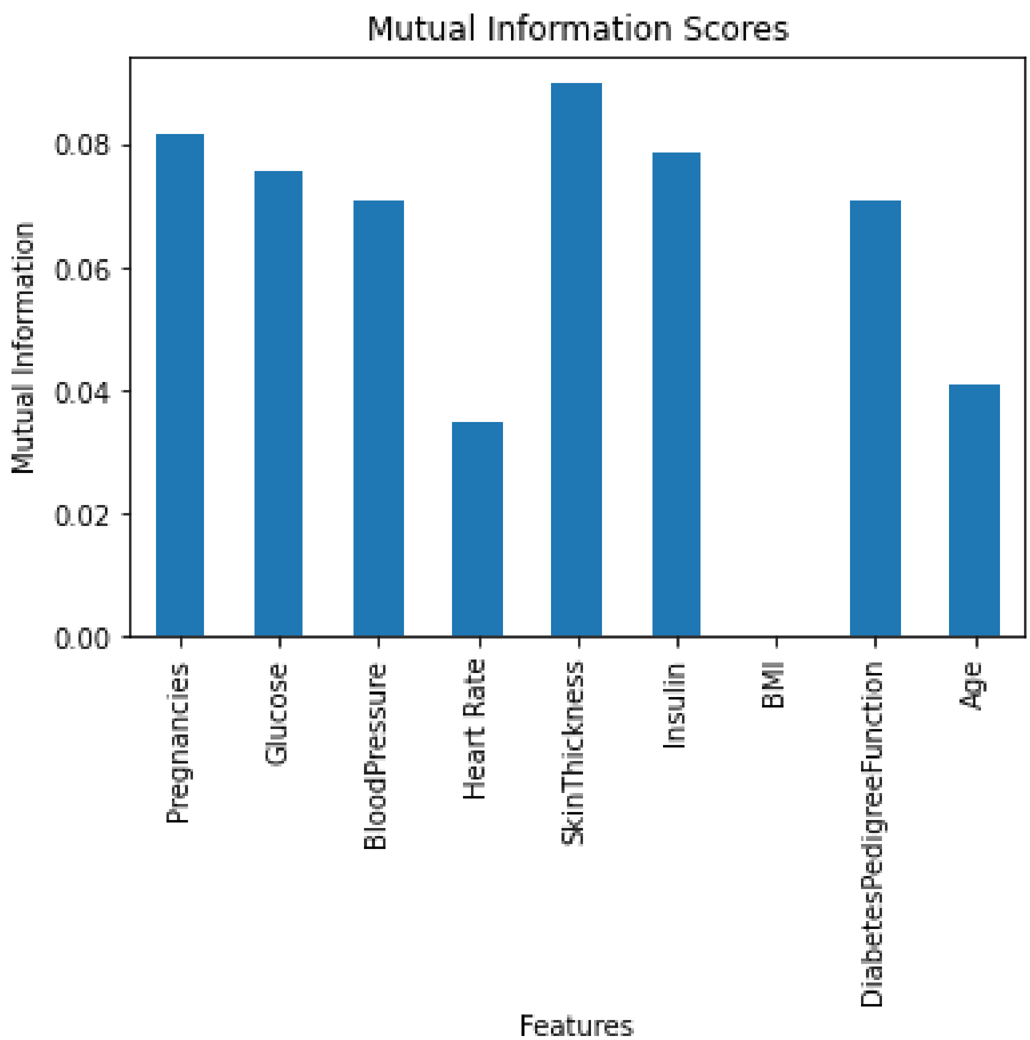
```
#mutual information
from sklearn.feature_selection import mutual_info_classif
import matplotlib.pyplot as plt

def plot_mutual_information(X, y, feature_names):
    mi_scores = mutual_info_classif(X, y)
    mi_scores_series = pd.Series(mi_scores, index=feature_names)
    mi_scores_series.plot(kind='bar')
    plt.title('Mutual Information Scores')
    plt.xlabel('Features')
    plt.ylabel('Mutual Information')
    return mi_scores_series.index.tolist()

kd_x =  kd_data.drop('class_label', axis=1).to_numpy()
kd_y = kd_data.to_numpy()[:, -1].reshape(-1,1)
print(kd_x.shape)
print(kd_y.shape)
```

```python
plot_mutual_information(kd_x,kd_y,kd_data.columns [:-1])
```

```
(329, 9)
(329, 1)
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/utils/validation.py:993:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

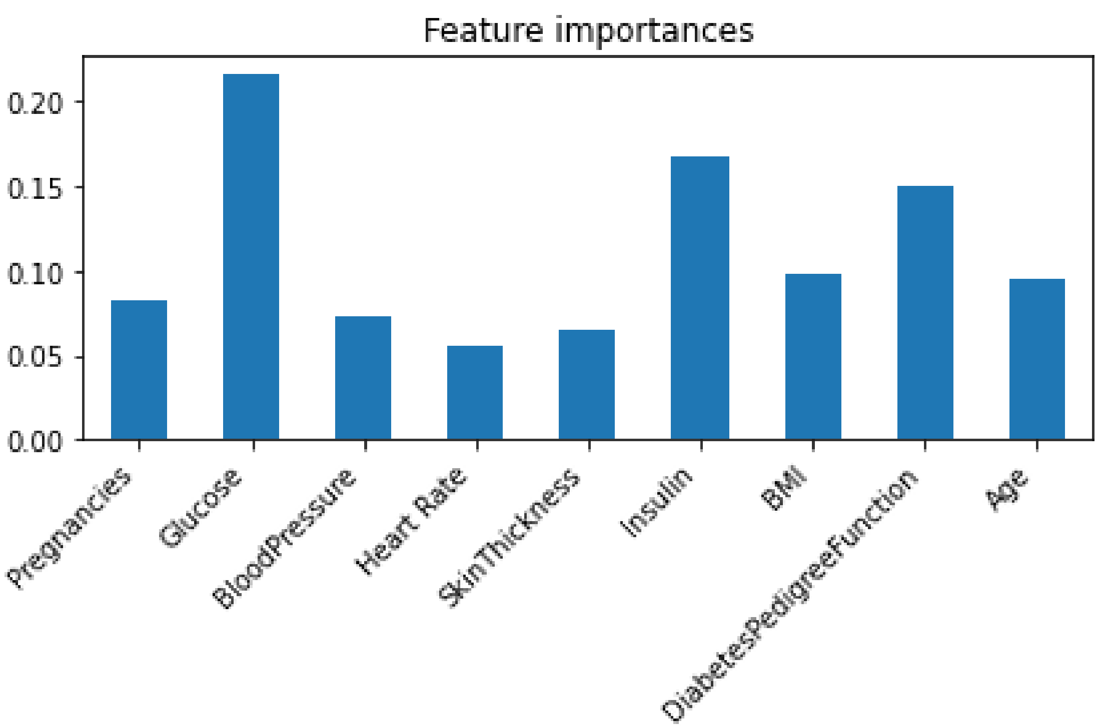
[38]:
```
['Pregnancies',
 'Glucose',
 'BloodPressure',
 'Heart Rate',
 'SkinThickness',
 'Insulin',
 'BMI',
 'DiabetesPedigreeFunction',
 'Age']
```



[39]:
```
#feature importance
from sklearn.ensemble import RandomForestClassifier
print(kd_features.shape)
feature_names = kd_data.columns[:-1]
print(feature_names)
y = kd_data.to_numpy()[:, -1]
print(y.shape)
forest = RandomForestClassifier(random_state=0)
forest.fit(kd_features, y)

importances = forest.feature_importances_
print(importances)
std = np.std([tree.feature_importances_ for tree in forest.estimators_], axis=0)
```

```
forest_importances = pd.Series(importances, index=feature_names)

kd_colors=['#A71930', '#DF4601', '#AB0003', '#003278', '#FF5910', '#0E3386',␣
↪'#BA0021', '#E81828', '#473729']
fig, ax = plt.subplots()
forest_importances.plot.bar(ax=ax)
plt.xticks(rotation=45,ha='right')
ax.set_title("Feature importances")
fig.tight_layout()
```

```
(329, 9)
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'Heart Rate',
       'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age'],
      dtype='object')
(329,)
[0.08160676 0.21521658 0.07359165 0.05590117 0.06448943 0.16692592
 0.09776524 0.15017762 0.09432563]
```



[40]:
```
#implement RFE
from sklearn.datasets import make_friedman1
from sklearn.feature_selection import RFE
from sklearn.feature_selection import RFECV
from sklearn.svm import SVR
```

```python
X = kd_features
y =  kd_data.to_numpy()[:, -1]
estimator = SVR(kernel="linear")
selector = RFE(estimator, n_features_to_select=8, step=1)
selector = selector.fit(X, y)
print(selector.support_)

print(selector.ranking_)

print(importances)

print('false indices:',np.where(selector.support_ == False)[0])
falseData = np.where(selector.support_ == False)[0];

print('final false features:',X.columns[falseData])
```

```
[ True  True  True False  True  True  True  True  True]
[1 1 1 2 1 1 1 1 1]
[0.08160676 0.21521658 0.07359165 0.05590117 0.06448943 0.16692592
 0.09776524 0.15017762 0.09432563]
false indices: [3]
final false features: Index(['Heart Rate'], dtype='object')
```

[40]:

```python

```

White_wine_analysis

February 19, 2023

[52]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[53]:
```
from google.colab import drive
drive.mount('/content/gdrive/', force_remount=True)

ww_columns = ['Alcohol', 'Malic acid', 'Ash', 'Alkalinity of ash', 'Magnesium',␣
 ↪'Total phenols', 'Flavanoids', 'Nonflavanoid phenols' , 'Proanthocyanins',␣
 ↪'Hue', 'class_label']
ww_data = pd.read_csv('/content/gdrive/MyDrive/White_wineQuality.csv',␣
 ↪names=ww_columns)
```

```
Mounted at /content/gdrive/
```

[54]:
```
ww_data.head()
```

[54]:

	Alcohol	Malic acid	Ash	Alkalinity of ash	Magnesium	Total phenols
0	0.28319	0.28082	0.04	0.054795	0.106840	0.225350
1	0.19469	0.34932	0.07	0.082192	0.128550	0.140850
2	0.30088	0.20548	0.17	0.075342	0.101840	0.070423
3	0.30088	0.20548	0.17	0.075342	0.101840	0.070423
4	0.24779	0.34247	0.05	0.068493	0.093489	0.323940

	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Hue	class_label
0	0.088339	0.46442	0.46457	0.13772	1
1	0.134280	0.38371	0.57480	0.13174	1
2	0.042403	0.52311	0.43307	0.16766	1
3	0.042403	0.52311	0.43307	0.16766	1
4	0.127210	0.44241	0.53543	0.14371	1

[55]:
```
ww_data.describe()
```

[55]:

	Alcohol	Malic acid	Ash	Alkalinity of ash	Magnesium
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000
mean	0.329172	0.279329	0.270976	0.112247	0.125987
std	0.154079	0.122644	0.194801	0.096570	0.078573

```
min        0.000000      0.000000      0.000000            0.000000      0.000000
25%        0.221240      0.184930      0.090000            0.068493      0.096828
50%        0.292040      0.273970      0.260000            0.089041      0.111850
75%        0.407080      0.356160      0.420000            0.116440      0.130220
max        1.000000      1.000000      1.000000            1.000000      1.000000

       Total phenols   Flavanoids  Nonflavanoid phenols  Proanthocyanins  \
count    1599.000000  1599.000000           1599.000000      1599.000000
mean        0.209506     0.142996              0.489853         0.449695
std         0.147326     0.116238              0.138461         0.121565
min         0.000000     0.000000              0.000000         0.000000
25%         0.084507     0.056537              0.405720         0.370080
50%         0.183100     0.113070              0.490100         0.448820
75%         0.281690     0.197880              0.569700         0.519690
max         1.000000     1.000000              1.000000         1.000000

               Hue  class_label
count  1599.000000  1599.000000
mean      0.196496     0.534709
std       0.101501     0.498950
min       0.000000     0.000000
25%       0.131740     0.000000
50%       0.173650     1.000000
75%       0.239520     1.000000
max       1.000000     1.000000
```

[56]:

```
ww_x =  ww_data.drop('class_label', axis=1).to_numpy()
ww_y = ww_data.to_numpy()[:, -1].reshape(-1,1)
print(ww_x.shape)
print(ww_y.shape)
```

```
(1599, 10)
(1599, 1)
```

[57]:

```
ww_feature_names = ww_data.columns[:-1]
ww_feature_number = len(ww_feature_names)
ww_features = ww_data.iloc[:,0:-1]
ww_data_cl_number = len(ww_data)
```

[58]:

```
# check duplicate in rows
ww_data.duplicated().sum()
print("is duplicate:",ww_data.duplicated().sum())

# check null input in rows
print("is null:",ww_data.isnull().values.any())

print("maxmium values",ww_data.max(axis=0))
```

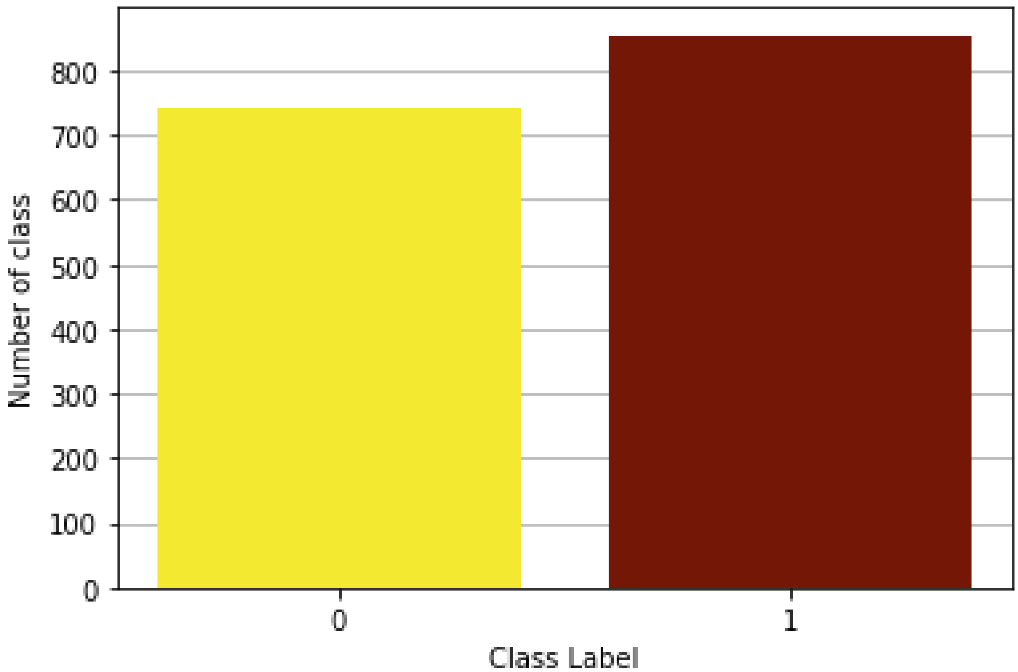
```
print("minimum values",ww_data.min(axis=0))
```

```
is duplicate: 244
is null: False
maxmium values Alcohol                  1.0
Malic acid              1.0
Ash                     1.0
Alkalinity of ash       1.0
Magnesium               1.0
Total phenols           1.0
Flavanoids              1.0
Nonflavanoid phenols    1.0
Proanthocyanins         1.0
Hue                     1.0
class_label             1.0
dtype: float64
minimum values Alcohol                  0.0
Malic acid              0.0
Ash                     0.0
Alkalinity of ash       0.0
Magnesium               0.0
Total phenols           0.0
Flavanoids              0.0
Nonflavanoid phenols    0.0
Proanthocyanins         0.0
Hue                     0.0
class_label             0.0
dtype: float64
```

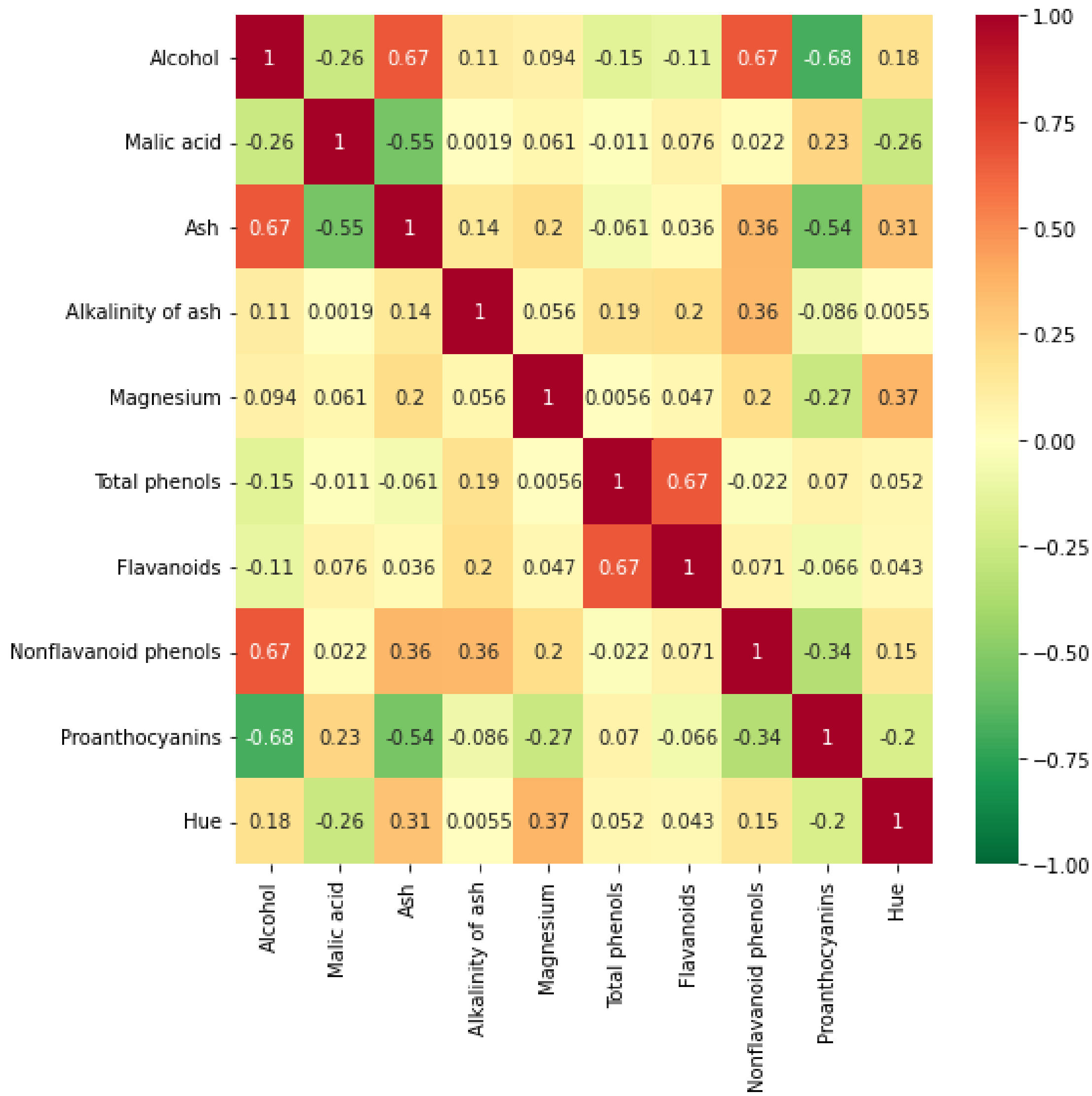
[59]:
```
# distribution of class 0 and class 1
label_counts = ww_data["class_label"].value_counts()
print(label_counts)
unique_labels = ww_data["class_label"].unique()

fig, ax = plt.subplots()
ax.grid(zorder=1, axis="y")
ax.bar(unique_labels, label_counts, zorder=2, color=['#751706', '#f2e930'])
ax.set_xticks([1,0])
ax.set_xticklabels(unique_labels)
ax.set_ylabel("Number of class")
ax.set_xlabel("Class Label")
```

```
1    855
0    744
Name: class_label, dtype: int64
```

[59]:
```
Text(0.5, 0, 'Class Label')
```



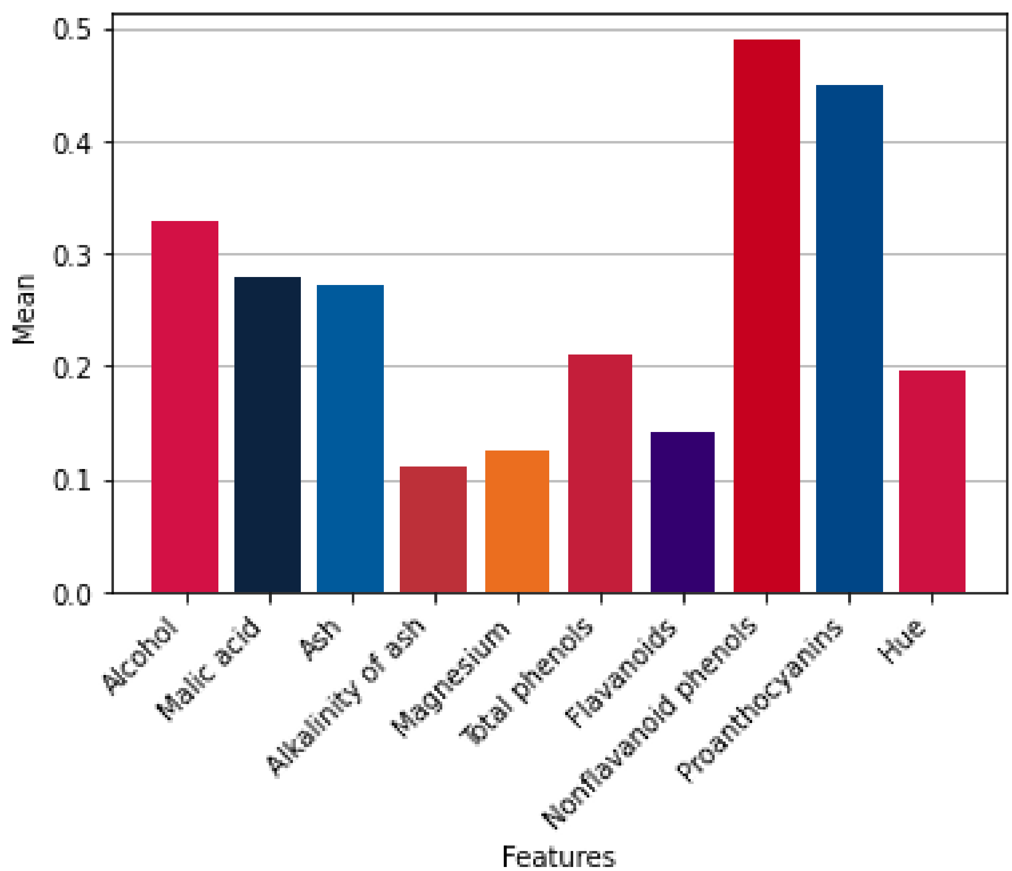
[60]:
```
# correlation between features
ww_corr = ww_features.corr()
plt.figure(figsize=(8,8))
sns.heatmap(ww_corr, cmap='RdYlGn_r', annot=True, vmin=-1, vmax=1)
plt.show()
```

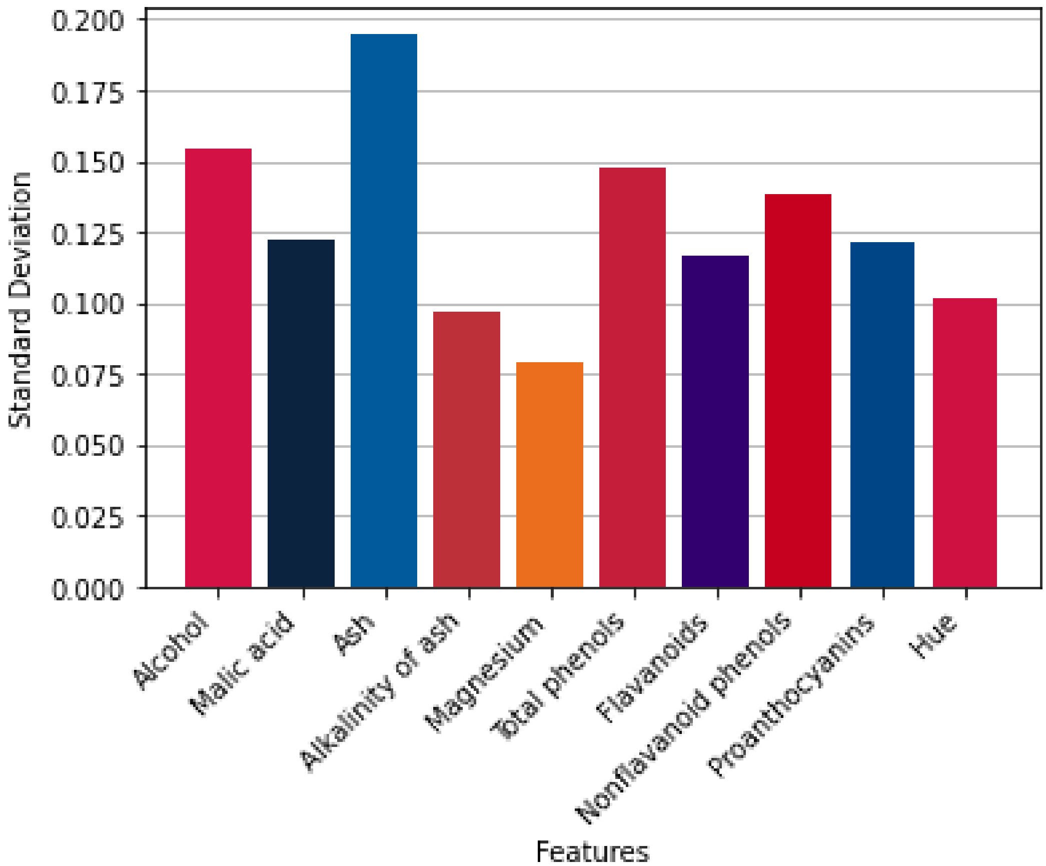


[61]:
```
# Compute mean of each feature
means = ww_features.mean()

# Compute standard deviation of each feature
std = ww_features.std()

ww_colors = ['#D31145', '#0C2340', '#005A9C', '#BD3039', '#EB6E1F', '#C41E3A',␣
↪'#33006F', '#C6011F', '#004687', '#CE1141']
# Plot mean and standard deviation of each attribute
fig, ax = plt.subplots()

ax.grid(zorder=1, axis="y")
ax.bar(ww_feature_names, means, zorder=2, color=ww_colors)
```

```
plt.xticks(rotation=45,ha='right')
ax.set_xlabel("Features")
ax.set_ylabel("Mean")


fig, ax = plt.subplots()
ax.grid(zorder=1, axis="y")
ax.bar(ww_feature_names, std, zorder=2, color=ww_colors)
ax.set_xlabel("Features")
plt.xticks(rotation=45,ha='right')
ax.set_ylabel("Standard Deviation")
```

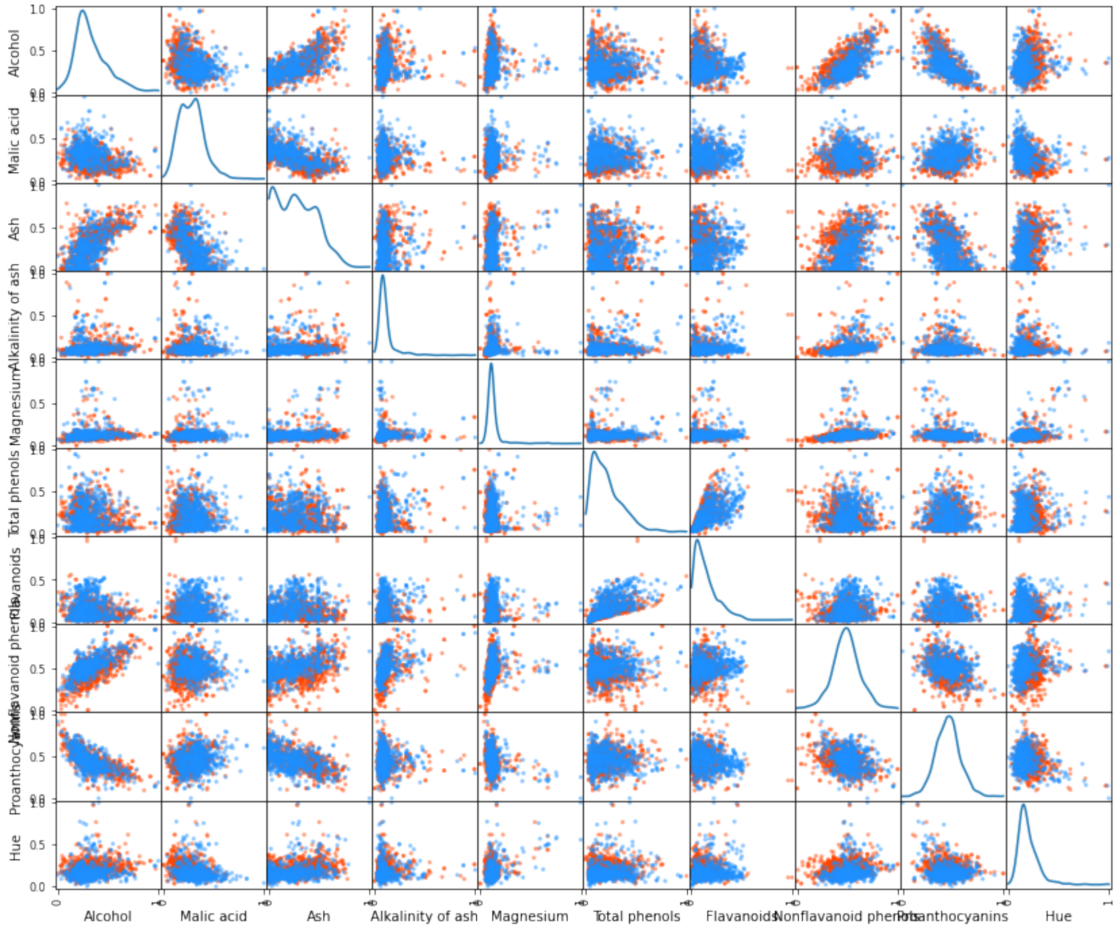
[61]: Text(0, 0.5, 'Standard Deviation')





[62]:

```
# correlation between features
color = ['dodgerblue', 'orangered']
colors = ww_data['class_label'].map(lambda x: color[x])
pd.plotting.scatter_matrix(ww_features, figsize= (14, 12), diagonal='kde',␣
 ↪color=colors);
```



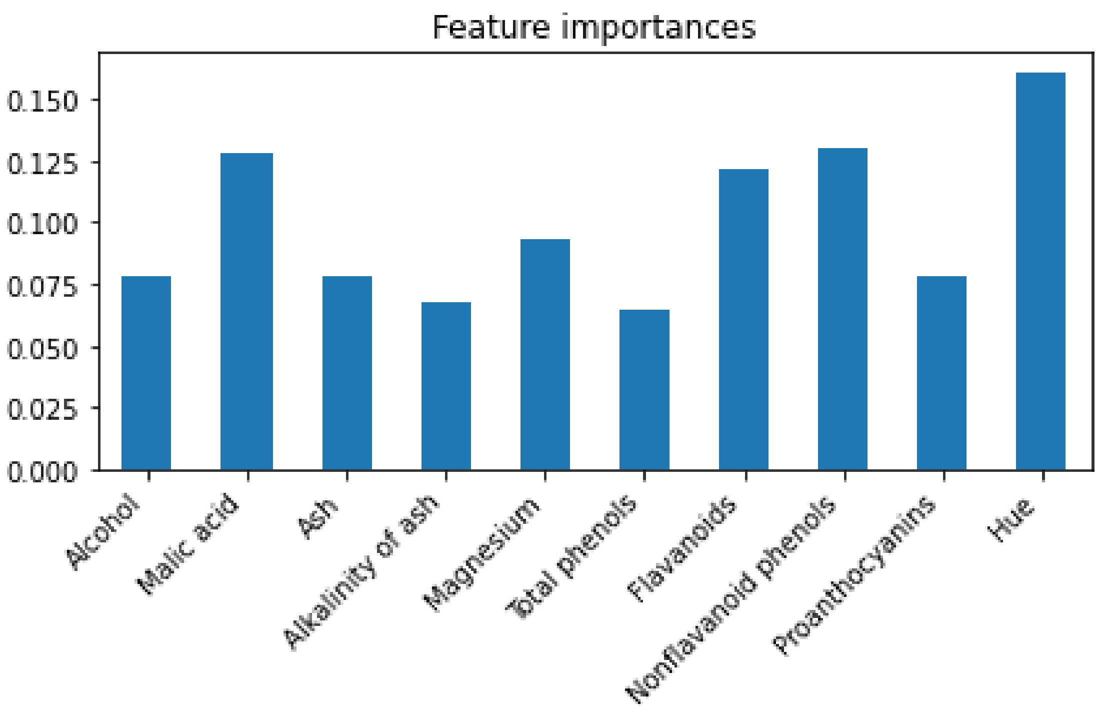
[63]:
```
ww_ndata = ww_data.to_numpy()
print(np.min(ww_ndata))
print(np.max(ww_ndata))
```

```
0.0
1.0
```

[68]:
```
from sklearn.ensemble import RandomForestClassifier

feature_names = ww_data.columns[:-1]
print(feature_names)
y = ww_data.to_numpy()[:, -1]
forest = RandomForestClassifier(random_state=0)
forest.fit(ww_features, y)

importances = forest.feature_importances_
print(importances)
std = np.std([tree.feature_importances_ for tree in forest.estimators_], axis=0)
```

```
forest_importances = pd.Series(importances, index=feature_names)
fig, ax = plt.subplots()
forest_importances.plot.bar(ax=ax)
#forest_importances.sort_values(ascending=False).plot.bar(ax=ax)
plt.xticks(rotation=45,ha='right')
ax.set_title("Feature importances")
fig.tight_layout()
```

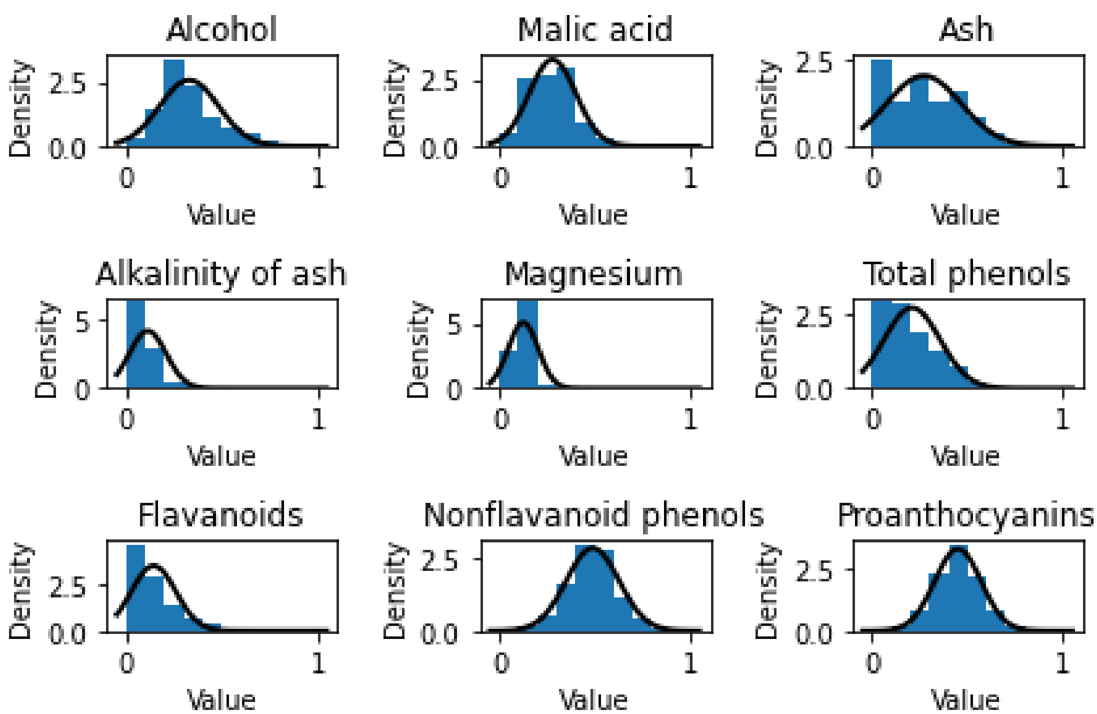
```
Index(['Alcohol', 'Malic acid', 'Ash', 'Alkalinity of ash', 'Magnesium',
       'Total phenols', 'Flavanoids', 'Nonflavanoid phenols',
       'Proanthocyanins', 'Hue'],
      dtype='object')
[0.07840397 0.12771454 0.07784699 0.06800892 0.09293432 0.06424689
 0.12183957 0.12987766 0.07837451 0.16075263]
```



[65]:
```
from scipy.stats import norm

def draw_distributon_graph(x_features,names):
  kd_normal_fig, axis = plt.subplots(3, 3)
  col = 0

  for x, ax in enumerate(axis.ravel()):
      ax.hist(x_features.loc[:, names[col]], density=True)
```

```
        mu, std = norm.fit(x_features.loc[:, names[col]])
        xmin, xmax = ax.get_xlim()
        x = np.linspace(xmin, xmax, 100)
        p = norm.pdf(x, mu, std)
        ax.plot(x, p, 'k', linewidth=2)
        ax.set_title(names[col])
        ax.set_xlabel("Value")
        ax.set_ylabel("Density")
        col += 1
    kd_normal_fig.tight_layout()

print(ww_feature_names)
draw_distributon_graph(ww_features,ww_feature_names)
```

```
Index(['Alcohol', 'Malic acid', 'Ash', 'Alkalinity of ash', 'Magnesium',
       'Total phenols', 'Flavanoids', 'Nonflavanoid phenols',
       'Proanthocyanins', 'Hue'],
      dtype='object')
```



[66]:

```
#impelementation of RFE
from sklearn.datasets import make_friedman1
from sklearn.feature_selection import RFE
from sklearn.feature_selection import RFECV
from sklearn.svm import SVR
```

```
X = ww_features
y =  ww_data.to_numpy()[:, -1]
estimator = SVR(kernel="linear")
selector = RFE(estimator, n_features_to_select=9, step=1)
selector = selector.fit(X, y)
print(selector.support_)

print(selector.ranking_)

estimator2 = SVR(kernel="linear")
selector2 = RFECV(estimator2, step=1, cv=5,min_features_to_select = 9)
selector2 = selector.fit(X, y)
print('2',selector2.support_)

print('2',selector2.ranking_)
```

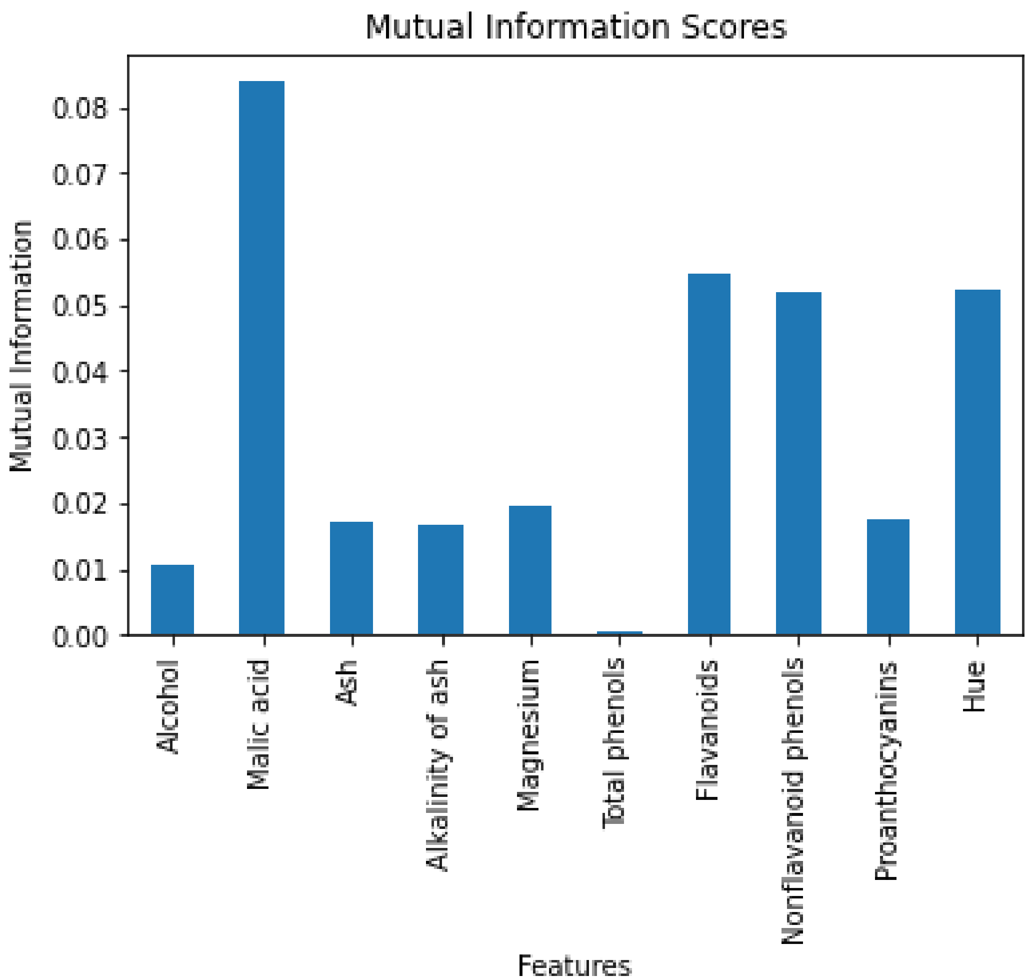
```
[ True  True False  True  True  True  True  True  True  True]
[1 1 2 1 1 1 1 1 1 1]
2 [ True  True False  True  True  True  True  True  True  True]
2 [1 1 2 1 1 1 1 1 1 1]
```

[67]:

```
#plot mutual information
from sklearn.feature_selection import mutual_info_classif
import matplotlib.pyplot as plt

from sklearn.feature_selection import mutual_info_classif
import matplotlib.pyplot as plt
import numpy as np

def plot_mutual_information(X, y, feature_names):
    np.random.seed(4)
    mi_scores = mutual_info_classif(X, y)
    mi_scores_series = pd.Series(mi_scores, index=feature_names)
    mi_scores_series.plot(kind='bar')
    plt.title('Mutual Information Scores')
    plt.xlabel('Features')
    plt.ylabel('Mutual Information')
    return mi_scores_series.sort_values(ascending=False).index.tolist()


plot_mutual_information(ww_x,ww_y,ww_columns[:-1])
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/utils/validation.py:993:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

[67]:
```
['Malic acid',
 'Flavanoids',
 'Hue',
 'Nonflavanoid phenols',
 'Magnesium',
 'Proanthocyanins',
 'Ash',
 'Alkalinity of ash',
 'Alcohol',
 'Total phenols']
```



kidney_disease_model

February 19, 2023

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm
from google.colab import drive
import random
import time
from enum import Enum
from sklearn.preprocessing import StandardScaler

drive.mount('/content/gdrive/', force_remount=True)

kd_columns = ['Pregnancies', 'Glucose', 'BloodPressure', 'Heart Rate',␣
 ↪'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction',␣
 ↪'Age','class_label']
kd_data_initial = pd.read_csv('/content/gdrive/MyDrive/kidney_disease.csv',␣
 ↪names = kd_columns )
print(kd_data_initial.shape)

ww_columns = ['Alcohol', 'Malic acid', 'Ash', 'Alkalinity of ash', 'Magnesium',␣
 ↪'Total phenols', 'Flavanoids', 'Nonflavanoid phenols' , 'Proanthocyanins',␣
 ↪'Hue', 'class_label']
ww_data_initial = pd.read_csv('/content/gdrive/MyDrive/white_wine_quality.
 ↪csv',names = ww_columns)
print(ww_data_initial.shape)
```

```
Mounted at /content/gdrive/
(330, 10)
(1599, 11)
```

[3]:
```
def shuffle_data(df):
    random.seed(0)  # Use a fixed seed for the random number generator
    df = df.sample(frac=1, random_state=0).reset_index(drop=True)
    return df
```

[4]:
```
kd_data = shuffle_data(kd_data_initial)
ww_data = shuffle_data(ww_data_initial)
```

[5]:
```
#get feature columns and lable column and convert it to array
kd_x =  kd_data.drop('class_label', axis=1).to_numpy()
kd_y = kd_data.to_numpy()[:, -1].reshape(-1,1)


ww_x = ww_data.drop('class_label', axis=1).to_numpy()
ww_y = ww_data.to_numpy()[:, -1].reshape(-1,1)
```

[6]:
```
#defined a dataframe for storing the result of model, an enum for different␣
↪learning types
model_data = pd.
↪DataFrame(columns=['model_name','description','learning_rate','iteration','weights','epsilo
                                    ␣
↪'is_max_reached','loss','accuracy_kfold','variable'])
learning_rate_type = Enum('lr_type', ['independent', 'iteration',␣
↪'iteration_plus_one','sample_size','ten_sample_size','hundred_sample_size'])
```

[7]:
```
#utility functions

def train_test_split(x, y, train_size=0.8):
    num_rows = x.shape[0]
    num_rows_train = int(num_rows * train_size )
    num_rows_test = num_rows - num_rows_train

    x_train = x[:num_rows_train, :]
    x_test =  x[num_rows_train:, :]
    y_train = y[:num_rows_train]
    y_test = y[num_rows_train:]

    return x_train, y_train, x_test, y_test


#convert feature to Gaussian distribution
def log_transform_normalize(df, index):
    df_copy = df.copy()
    df_copy.iloc[:, index] = df_copy.iloc[:, index].apply(lambda x: np.nan if x␣
↪<= 0 else x)
    #add small values to avoid NaN
    df_copy.iloc[:, index] = np.log(df_copy.iloc[:, index] + 1e-10)
    mean_val = df_copy.iloc[:, index].mean()
    df_copy.iloc[:, index] = df_copy.iloc[:, index].fillna(mean_val)
    scaler = StandardScaler()
    df_copy.iloc[:, index] = scaler.fit_transform(df_copy.iloc[:, index].values.
↪reshape(-1, 1))
```

```
    return df_copy


#convert feature to the power of feature
def power_n_feature(df, index, power_number):
    df_copy = df.copy()
    df_copy.iloc[:, index] = df_copy.iloc[:, index].apply(lambda x: x **␣
 ↪power_number)
    return df_copy
```

[8]:
```
class Logistic_Regression:

    def __init__(self, learning_rate , learning_rate_type , max_iterations ,␣
 ↪epsilon):
        self.x = []
        self.y = []
        self.weights = []
        self.learning_rate = learning_rate
        self.learning_rate_type = learning_rate_type
        self.max_iterations = max_iterations
        self.epsilon = epsilon


    def fit(self,x, y,is_add_bias = False):
        self.x = x;
        self.y = y;
        loss_list = []
        t_start = time.time()
        n, m = self.x.shape
        is_max_reached = False

        #add bias/dummy feature
        #initialize weights with zero
        if(is_add_bias == True):
          bias = np.ones((n,1), dtype=np.double)
          self.x = np.append(self.x, bias, axis = 1)
          self.weights = np.zeros(((m+1),1))
        else:
          self.weights = np.zeros(((m),1))


        #define initial norm value
        norm_weights = 1e8
        iteration = 1

        while (iteration < self.max_iterations) & (norm_weights > self.epsilon):
          #if(iteration%10000 == 0):
```

```
            #print('iteration number:',iteration)

          if(self.learning_rate_type == learning_rate_type.iteration):
              self.learning_rate = (1/ iteration)
          elif(self.learning_rate_type == learning_rate_type.
↪iteration_plus_one):
              self.learning_rate = (1/ (1 + iteration))
          elif(self.learning_rate_type == learning_rate_type.sample_size):
              self.learning_rate = (1/ (1 + n))
          elif(self.learning_rate_type == learning_rate_type.ten_sample_size):
              self.learning_rate = (1/ (10 * n))
          elif(self.learning_rate_type == learning_rate_type.
↪hundred_sample_size):
              self.learning_rate = (1/ (100 * n))

          #if(iteration == 15):
           # print("learning rate type:",self.learning_rate_type)
           # print("learning rate:",self.learning_rate)

          if iteration % 100 == 0:
            loss_model = self.cross_entropy_loss(self.x, self.y);
            loss_list.append((iteration, loss_model))

          # Store current weights before updating
          weight_previous = self.weights

          # Compute gradient
          gradient =  np.sum(
          self.x * (self.y - self.sigmoid(np.dot(self.x, weight_previous))),␣
↪axis=0
          ).reshape(-1, 1)

          # Update weights
          self.weights = weight_previous + self.learning_rate * gradient

          # Compute change in weights
          norm_weights = np.linalg.norm(self.weights - weight_previous) ** 2
          iteration += 1

        if(iteration == self.max_iterations):
         print (f"**************failed to reach minimum in {self.
↪max_iterations} iterations")
         is_max_reached = True


        t_end = time.time()
```

```
        #time elapsed for model training
        elapsed_time = round(t_end - t_start,3)
        return iteration,self.weights, elapsed_time,is_max_reached,loss_list


      ############################################ end fit
    # Decision boundary(threshold)
    def predict(self):
      decision_boundary = 0.5
      y_predict = self.sigmoid(np.dot(self.x, self.weights))
      y_pred = np.where(y_predict < decision_boundary, 0, 1)
      return y_pred

    #compute accuracy of model
    def accu_eval(self, y_pred):
      accuracy = np.count_nonzero(self.y == y_pred) / len(self.y)
      return accuracy


    def sigmoid(self, arg):
        return 1 / (1 + np.exp(-arg))

    #compute cross entropy loss
    def cross_entropy_loss(self, x_data, y_data):

        y_pred_0 = self.sigmoid(np.dot(x_data,self.weights))
        y_pred_1 = 1 - y_pred_0
        # Replace small values to avoid NAN (log0)
        y_pred_0 = np.where(y_pred_0 < 1e-6, 1e-6, y_pred_0)
        y_pred_1 = np.where(y_pred_1 < 1e-6, 1e-6, y_pred_1)

        loss_0 = y_data * np.log(y_pred_0)
        loss_1 = (1-y_data) * np.log(y_pred_1)
        loss = -np.sum(loss_0 + loss_1)
        return loss
```

[9]:
```
#k fold cross validation function
def kfold_cross_validation(lgr_model , k = 10, x_train_initial = []␣
 ↪,y_train_initial= [] ):

  partition_size = int(len(x_train_initial)/k)

  model_accuracy_list = []
  model_loss_list = []

  for i in range(k):
    print("i======>",i)
```

```
        # Split data
        i_start = partition_size * i
        i_end = partition_size*(i+1)

        if i != (k-1):
          x_train_fold = np.concatenate((x_train_initial[:i_start,:],␣
 ↪x_train_initial[i_end:,:]),axis=0)
          y_train_fold = np.concatenate((y_train_initial[:i_start,:],␣
 ↪y_train_initial[i_end:,:]),axis=0)
          x_validation_fold = x_train_initial[i_start:i_end,:]
          y_validation_fold = y_train_initial[i_start:i_end,:]

        else:
          # For final partition
          x_train_fold = lgr_model.x[:i_start,:]
          y_train_fold = lgr_model.y[:i_start,:]
          x_validation_fold = lgr_model.x[i_start:,:]
          y_validation_fold = lgr_model.y[i_start:,:]


        iteration, weight_store , epalsed_time_one , is_max_reached , loss_list =␣
 ↪lgr_model.fit(x_train_fold,y_train_fold,False)
        y_predict = lgr_model.predict()

        model_accuracy = lgr_model.accu_eval(y_predict)
        model_accuracy_list.append(model_accuracy)
        cross_entropy = lgr_model.
 ↪cross_entropy_loss(x_validation_fold,y_validation_fold)
        model_loss_list.append(cross_entropy)
        #print("model_accuracy:",model_accuracy)
        #return model_accuracy
    return np.mean(model_accuracy_list),np.mean(model_loss_list)
```

[10]:
```
#define a function that gets the weights and runs k-fold algorithm,then stores␣
 ↪it in model data
def run_model(model_name , description,learning_rate , learning_rate_type,␣
 ↪max_iterations , epsilon,  x_train ,   y_train , model_data , variable):

    model = Logistic_Regression(learning_rate = learning_rate, learning_rate_type␣
 ↪= learning_rate_type , max_iterations = max_iterations,epsilon = epsilon)
    model_iteration_num , model_weights  , model_elapsed_time ,␣
 ↪is_max_iteration_reached, loss_list  = model.fit(x_train,y_train, True)
```

```
  model_accuracy_kfold,model_loss_kfold = kfold_cross_validation(model,␣
↪k=10,x_train_initial = model.x , y_train_initial =  model.y)

  final_learning_rate = model.learning_rate if  learning_rate_type ==␣
↪learning_rate_type.independent else learning_rate_type

  model_data = model_data.append({'model_name':model_name,'description':
↪description,'learning_rate' : final_learning_rate,
                                 'iteration' : model_iteration_num,'weights' :␣
↪model_weights,'epsilon':model.epsilon,
                                 'elapsed_time':model_elapsed_time,␣
↪'is_max_reached': is_max_iteration_reached, 'loss':model_loss_kfold,
                             'accuracy_kfold':model_accuracy_kfold ,␣
↪'variable':variable}, ignore_index = True)
  return model_data
```

[11]:
```
kd_x_train, kd_y_train, kd_x_test, kd_y_test = train_test_split(kd_x,kd_y,1)
print(kd_x_train.shape)
print(kd_y_train.shape)

ww_x_train, ww_y_train, ww_x_test, ww_y_test = train_test_split(ww_x,ww_y,1)
```

```
(330, 9)
(330, 1)
```

[12]:
```
#function sort dataframe by accuracy and delete dataframe
def show_sorted_model(data):
  return data.sort_values(by=['accuracy_kfold'], ascending=False)
def delete_model(data):
  return data.drop(model_data.index,inplace=True)
def delete_last_model(data,number):
  return data.drop(data.tail(number).index,inplace=True)
```

[13]:
```
#delete_model(model_data)
show_sorted_model(model_data)
#print(kd_x_train.shape)
#delete_model(model_data)
```

[13]:
```
Empty DataFrame
Columns: [model_name, description, learning_rate, iteration, weights, epsilon,
elapsed_time, is_max_reached, loss, accuracy_kfold, variable]
Index: []
```

[14]:
```
#train whole model
model_data = run_model('kd','whole model',learning_rate = 0.01,␣
↪learning_rate_type = learning_rate_type.independent ,max_iterations = 150000,
```

```
                        epsilon = 1e-6,x_train = kd_x_train , y_train =␣
↪kd_y_train , model_data = model_data,variable = 'all features')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[14]:

```
  model_name  description  learning_rate iteration  \
0         kd  whole model           0.01       956

                                             weights   epsilon  elapsed_time  \
0  [[2.5819277747484066], [6.780007402086465], [-…  0.000001         0.049

  is_max_reached       loss  accuracy_kfold      variable
0          False  15.855793        0.749495  all features
```

[15]:

```
#explore different constant learning rates
learning_rates = [1/2,1/4,1/8,0.05]
for i in range(len(learning_rates)):
  title = 'whole model-lr:' + str(learning_rates[i])
  model_data = run_model('kd',title,learning_rate =  learning_rates[i],␣
↪learning_rate_type = learning_rate_type.independent ,max_iterations = 150000,
                        epsilon = 1e-6,x_train = kd_x_train , y_train =␣
↪kd_y_train , model_data = model_data,variable = 'learning rate')
show_sorted_model(model_data)
```

```
**************failed to reach minimum in 150000 iterations
i======> 0
**************failed to reach minimum in 150000 iterations
i======> 1
**************failed to reach minimum in 150000 iterations
i======> 2
**************failed to reach minimum in 150000 iterations
i======> 3
**************failed to reach minimum in 150000 iterations
i======> 4
**************failed to reach minimum in 150000 iterations
i======> 5
**************failed to reach minimum in 150000 iterations
```

```
i======> 6
**************failed to reach minimum in 150000 iterations
i======> 7
**************failed to reach minimum in 150000 iterations
i======> 8
**************failed to reach minimum in 150000 iterations
i======> 9
**************failed to reach minimum in 150000 iterations
**************failed to reach minimum in 150000 iterations
i======> 0
**************failed to reach minimum in 150000 iterations
i======> 1
**************failed to reach minimum in 150000 iterations
i======> 2
**************failed to reach minimum in 150000 iterations
i======> 3
**************failed to reach minimum in 150000 iterations
i======> 4
**************failed to reach minimum in 150000 iterations
i======> 5
**************failed to reach minimum in 150000 iterations
i======> 6
**************failed to reach minimum in 150000 iterations
i======> 7
**************failed to reach minimum in 150000 iterations
i======> 8
**************failed to reach minimum in 150000 iterations
i======> 9
**************failed to reach minimum in 150000 iterations
**************failed to reach minimum in 150000 iterations
i======> 0
**************failed to reach minimum in 150000 iterations
i======> 1
**************failed to reach minimum in 150000 iterations
i======> 2
**************failed to reach minimum in 150000 iterations
i======> 3
**************failed to reach minimum in 150000 iterations
i======> 4
**************failed to reach minimum in 150000 iterations
i======> 5
**************failed to reach minimum in 150000 iterations
i======> 6
**************failed to reach minimum in 150000 iterations
i======> 7
**************failed to reach minimum in 150000 iterations
i======> 8
**************failed to reach minimum in 150000 iterations
```

```
i======> 9
**************failed to reach minimum in 150000 iterations
**************failed to reach minimum in 150000 iterations
i======> 0
**************failed to reach minimum in 150000 iterations
i======> 1
**************failed to reach minimum in 150000 iterations
i======> 2
**************failed to reach minimum in 150000 iterations
i======> 3
**************failed to reach minimum in 150000 iterations
i======> 4
**************failed to reach minimum in 150000 iterations
i======> 5
**************failed to reach minimum in 150000 iterations
i======> 6
**************failed to reach minimum in 150000 iterations
i======> 7
**************failed to reach minimum in 150000 iterations
i======> 8
**************failed to reach minimum in 150000 iterations
i======> 9
**************failed to reach minimum in 150000 iterations
```

[15]:

```
  model_name           description  learning_rate  iteration  \
0         kd           whole model          0.010        956
4         kd   whole model-lr:0.05          0.050     150000
2         kd   whole model-lr:0.25          0.250     150000
1         kd    whole model-lr:0.5          0.500     150000
3         kd  whole model-lr:0.125          0.125     150000

                                             weights   epsilon  elapsed_time  \
0  [[2.5819277747484066], [6.780007402086465], [-…  0.000001         0.049
4  [[9.970728371948313], [26.491694165903258], [-…  0.000001         6.472
2  [[47.64433276868854], [130.6790054571249], [-1…  0.000001         6.500
1  [[95.81933692733199], [262.7825656994754], [-3…  0.000001         8.592
3  [[24.070810464619214], [65.46666343519061], [-…  0.000001         8.383

  is_max_reached        loss  accuracy_kfold       variable
0          False   15.855793        0.749495   all features
4           True   46.838322        0.688552  learning rate
2           True  106.221172        0.683502  learning rate
1           True  118.189617        0.683165  learning rate
3           True   87.970034        0.683165  learning rate
```

[16]:

```
#explore different dependent learning rates
```

```
learning_rates_types = [learning_rate_type.iteration, learning_rate_type.
 ↪iteration_plus_one,learning_rate_type.sample_size,
                        learning_rate_type.ten_sample_size,learning_rate_type.
 ↪hundred_sample_size]
for i in range(len(learning_rates_types)):
  title = 'whole model-lr:' + str(learning_rates_types[i])
  model_data = run_model('kd',title,learning_rate =  0, learning_rate_type =␣
 ↪learning_rates_types[i] ,max_iterations = 150000,
                       epsilon = 1e-6,x_train = kd_x_train , y_train =␣
 ↪kd_y_train , model_data = model_data,variable = 'learning rate')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
```

```
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[16]:

```
  model_name                                description  \
6         kd   whole model-lr:lr_type.iteration_plus_one
5         kd            whole model-lr:lr_type.iteration
0         kd                                whole model
7         kd          whole model-lr:lr_type.sample_size
8         kd      whole model-lr:lr_type.ten_sample_size
4         kd                       whole model-lr:0.05
2         kd                       whole model-lr:0.25
1         kd                        whole model-lr:0.5
3         kd                      whole model-lr:0.125
9         kd  whole model-lr:lr_type.hundred_sample_size

                  learning_rate iteration  \
6   lr_type.iteration_plus_one      1476
5            lr_type.iteration      2220
0                         0.01       956
7          lr_type.sample_size      2048
8      lr_type.ten_sample_size      3516
4                         0.05    150000
2                         0.25    150000
1                          0.5    150000
3                        0.125    150000
9  lr_type.hundred_sample_size         2

                                             weights   epsilon  elapsed_time  \
6  [[3.175241989157885], [9.517996847051375], [-1…  0.000001         0.105
5  [[3.6602142620593963], [11.56162794637247], [-…  0.000001         0.162
0  [[2.5819277747484066], [6.780007402086465], [-…  0.000001         0.049
7  [[2.5011580589018196], [6.514304926750582], [-…  0.000001         0.155
8  [[1.8917011563673924], [3.7425174840678586], […  0.000001         0.266
4  [[9.970728371948313], [26.491694165903258], [-…  0.000001         6.472
```

```
2  [[47.64433276868854], [130.6790054571249], [-1…  0.000001         6.500
1  [[95.81933692733199], [262.7825656994754], [-3…  0.000001         8.592
3  [[24.070810464619214], [65.46666343519061], [-…  0.000001         8.383
9  [[0.0003065942121212121], [0.00038259484848484…  0.000001         0.000

  is_max_reached        loss  accuracy_kfold       variable
6          False   16.436353        0.751515  learning rate
5          False   17.234261        0.750505  learning rate
0          False   15.855793        0.749495   all features
7          False   15.849570        0.749495  learning rate
8          False   16.462365        0.741414  learning rate
4           True   46.838322        0.688552  learning rate
2           True  106.221172        0.683502  learning rate
1           True  118.189617        0.683165  learning rate
3           True   87.970034        0.683165  learning rate
9          False   20.585959        0.516835  learning rate
```

[17]:
```python
#explore different epsilons
epsilon_list = [1e-2,1e-3,1e-4,1e-5,1e-6,1e-7,1e-8,1e-9]
for i in range(len(epsilon_list)):
  title = 'whole model-epsilon:' + str(epsilon_list[i])
  model_data = run_model('kd',title,learning_rate =  0, learning_rate_type =␣
 ↪learning_rate_type.iteration_plus_one ,max_iterations = 150000,
                       epsilon = epsilon_list[i],x_train = kd_x_train , y_train␣
 ↪= kd_y_train , model_data = model_data,variable = 'epsilon')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
```

```
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[17]:
```
   model_name                                    description  \
16         kd                     whole model-epsilon:1e-08
14         kd                     whole model-epsilon:1e-06
6          kd   whole model-lr:lr_type.iteration_plus_one
15         kd                     whole model-epsilon:1e-07
17         kd                     whole model-epsilon:1e-09
5          kd            whole model-lr:lr_type.iteration
13         kd                     whole model-epsilon:1e-05
0          kd                                  whole model
7          kd          whole model-lr:lr_type.sample_size
12         kd                    whole model-epsilon:0.0001
11         kd                     whole model-epsilon:0.001
8          kd      whole model-lr:lr_type.ten_sample_size
10         kd                      whole model-epsilon:0.01
4          kd                          whole model-lr:0.05
2          kd                          whole model-lr:0.25
1          kd                           whole model-lr:0.5
3          kd                         whole model-lr:0.125
9          kd  whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
16  lr_type.iteration_plus_one      9021
14  lr_type.iteration_plus_one      1476
6   lr_type.iteration_plus_one      1476
15  lr_type.iteration_plus_one      3643
17  lr_type.iteration_plus_one     22164
5            lr_type.iteration      2220
13  lr_type.iteration_plus_one       636
0                         0.01       956
7          lr_type.sample_size      2048
12  lr_type.iteration_plus_one       314
11  lr_type.iteration_plus_one       165
8      lr_type.ten_sample_size      3516
10  lr_type.iteration_plus_one        84
```

```
4                        0.05    150000
2                        0.25    150000
1                         0.5    150000
3                       0.125    150000
9   lr_type.hundred_sample_size         2

                                                weights       epsilon  \
16  [[2.96533714092659], [8.33217514426845], [-1.0…  1.000000e-08
14  [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
6   [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
15  [[3.0601381991207917], [8.83862515342757], [-1…  1.000000e-07
17  [[2.887385852612343], [7.9553323804450375], [-…  1.000000e-09
5   [[3.6602142620593963], [11.56162794637247], [-…  1.000000e-06
13  [[3.3327577160627104], [10.39377495841507], [-…  1.000000e-05
0   [[2.5819277747484066], [6.780007402086465], [-…  1.000000e-06
7   [[2.5011580589018196], [6.514304926750582], [-…  1.000000e-06
12  [[3.6109084400585387], [11.42702526345321], [-…  1.000000e-04
11  [[4.254111597053624], [12.705390512862264], [-…  1.000000e-03
8   [[1.8917011563673924], [3.7425174840678586], […  1.000000e-06
10  [[5.958742925531525], [14.327917968306403], [-…  1.000000e-02
4   [[9.970728371948313], [26.491694165903258], [-…  1.000000e-06
2   [[47.64433276868854], [130.6790054571249], [-1…  1.000000e-06
1   [[95.81933692733199], [262.7825656994754], [-3…  1.000000e-06
3   [[24.070810464619214], [65.46666343519061], [-…  1.000000e-06
9   [[0.0003065942121212121], [0.00038259484848484…  1.000000e-06

    elapsed_time is_max_reached        loss  accuracy_kfold       variable
16         0.413          False   16.097559        0.751515        epsilon
14         0.072          False   16.436353        0.751515        epsilon
6          0.105          False   16.436353        0.751515  learning rate
15         0.166          False   16.225842        0.751178        epsilon
17         0.940          False   16.019264        0.750842        epsilon
5          0.162          False   17.234261        0.750505  learning rate
13         0.027          False   16.775910        0.749495        epsilon
0          0.049          False   15.855793        0.749495   all features
7          0.155          False   15.849570        0.749495  learning rate
12         0.013          False   17.369442        0.747475        epsilon
11         0.009          False   18.607920        0.743771        epsilon
8          0.266          False   16.462365        0.741414  learning rate
10         0.009          False   21.366778        0.737037        epsilon
4          6.472           True   46.838322        0.688552  learning rate
2          6.500           True  106.221172        0.683502  learning rate
1          8.592           True  118.189617        0.683165  learning rate
3          8.383           True   87.970034        0.683165  learning rate
9          0.000          False   20.585959        0.516835  learning rate
```

[18]:

```
show_sorted_model(model_data)
```

[18]:
```
   model_name                                description  \
16         kd                whole model-epsilon:1e-08
14         kd                whole model-epsilon:1e-06
6          kd   whole model-lr:lr_type.iteration_plus_one
15         kd                whole model-epsilon:1e-07
17         kd                whole model-epsilon:1e-09
5          kd           whole model-lr:lr_type.iteration
13         kd                whole model-epsilon:1e-05
0          kd                              whole model
7          kd         whole model-lr:lr_type.sample_size
12         kd               whole model-epsilon:0.0001
11         kd                whole model-epsilon:0.001
8          kd     whole model-lr:lr_type.ten_sample_size
10         kd                 whole model-epsilon:0.01
4          kd                      whole model-lr:0.05
2          kd                      whole model-lr:0.25
1          kd                       whole model-lr:0.5
3          kd                     whole model-lr:0.125
9          kd  whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
16   lr_type.iteration_plus_one      9021
14   lr_type.iteration_plus_one      1476
6    lr_type.iteration_plus_one      1476
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
5             lr_type.iteration      2220
13   lr_type.iteration_plus_one       636
0                          0.01       956
7           lr_type.sample_size      2048
12   lr_type.iteration_plus_one       314
11   lr_type.iteration_plus_one       165
8       lr_type.ten_sample_size      3516
10   lr_type.iteration_plus_one        84
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
3                         0.125    150000
9   lr_type.hundred_sample_size         2

                                              weights       epsilon  \
16  [[2.96533714092659], [8.33217514426845], [-1.0…  1.000000e-08
14  [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
6   [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
15  [[3.0601381991207917], [8.83862515342757], [-1…  1.000000e-07
17  [[2.887385852612343], [7.9553323804450375], [-…  1.000000e-09
5   [[3.6602142620593963], [11.56162794637247], [-…  1.000000e-06
```

```
13  [[3.3327577160627104], [10.39377495841507], [-…  1.000000e-05
0   [[2.5819277747484066], [6.780007402086465], [-…  1.000000e-06
7   [[2.5011580589018196], [6.514304926750582], [-…  1.000000e-06
12  [[3.6109084400585387], [11.42702526345321], [-…  1.000000e-04
11  [[4.254111597053624], [12.705390512862264], [-…  1.000000e-03
8   [[1.8917011563673924], [3.7425174840678586], […  1.000000e-06
10  [[5.958742925531525], [14.327917968306403], [-…  1.000000e-02
4   [[9.970728371948313], [26.491694165903258], [-…  1.000000e-06
2   [[47.64433276868854], [130.6790054571249], [-1…  1.000000e-06
1   [[95.81933692733199], [262.7825656994754], [-3…  1.000000e-06
3   [[24.070810464619214], [65.46666343519061], [-…  1.000000e-06
9   [[0.0003065942121212121], [0.00038259484848484…  1.000000e-06

    elapsed_time is_max_reached        loss  accuracy_kfold       variable
16         0.413          False   16.097559        0.751515        epsilon
14         0.072          False   16.436353        0.751515        epsilon
6          0.105          False   16.436353        0.751515  learning rate
15         0.166          False   16.225842        0.751178        epsilon
17         0.940          False   16.019264        0.750842        epsilon
5          0.162          False   17.234261        0.750505  learning rate
13         0.027          False   16.775910        0.749495        epsilon
0          0.049          False   15.855793        0.749495   all features
7          0.155          False   15.849570        0.749495  learning rate
12         0.013          False   17.369442        0.747475        epsilon
11         0.009          False   18.607920        0.743771        epsilon
8          0.266          False   16.462365        0.741414  learning rate
10         0.009          False   21.366778        0.737037        epsilon
4          6.472           True   46.838322        0.688552  learning rate
2          6.500           True  106.221172        0.683502  learning rate
1          8.592           True  118.189617        0.683165  learning rate
3          8.383           True   87.970034        0.683165  learning rate
9          0.000          False   20.585959        0.516835  learning rate
```

[19]:
```
show_sorted_model(model_data)
```

[19]:
```
   model_name                               description  \
16         kd              whole model-epsilon:1e-08
14         kd              whole model-epsilon:1e-06
6          kd  whole model-lr:lr_type.iteration_plus_one
15         kd              whole model-epsilon:1e-07
17         kd              whole model-epsilon:1e-09
5          kd            whole model-lr:lr_type.iteration
13         kd              whole model-epsilon:1e-05
0          kd                               whole model
7          kd          whole model-lr:lr_type.sample_size
12         kd             whole model-epsilon:0.0001
11         kd              whole model-epsilon:0.001
```

```
8          kd      whole model-lr:lr_type.ten_sample_size
10         kd                    whole model-epsilon:0.01
4          kd                         whole model-lr:0.05
2          kd                         whole model-lr:0.25
1          kd                          whole model-lr:0.5
3          kd                        whole model-lr:0.125
9          kd  whole model-lr:lr_type.hundred_sample_size

                  learning_rate iteration  \
16   lr_type.iteration_plus_one      9021
14   lr_type.iteration_plus_one      1476
6    lr_type.iteration_plus_one      1476
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
5             lr_type.iteration      2220
13   lr_type.iteration_plus_one       636
0                          0.01       956
7           lr_type.sample_size      2048
12   lr_type.iteration_plus_one       314
11   lr_type.iteration_plus_one       165
8       lr_type.ten_sample_size      3516
10   lr_type.iteration_plus_one        84
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
3                         0.125    150000
9   lr_type.hundred_sample_size         2

                                              weights       epsilon  \
16  [[2.96533714092659], [8.33217514426845], [-1.0…  1.000000e-08
14  [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
6   [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
15  [[3.0601381991207917], [8.83862515342757], [-1…  1.000000e-07
17  [[2.887385852612343], [7.9553323804450375], [-…  1.000000e-09
5   [[3.6602142620593963], [11.56162794637247], [-…  1.000000e-06
13  [[3.3327577160627104], [10.39377495841507], [-…  1.000000e-05
0   [[2.5819277747484066], [6.780007402086465], [-…  1.000000e-06
7   [[2.5011580589018196], [6.514304926750582], [-…  1.000000e-06
12  [[3.6109084400585387], [11.42702526345321], [-…  1.000000e-04
11  [[4.254111597053624], [12.705390512862264], [-…  1.000000e-03
8   [[1.8917011563673924], [3.7425174840678586], […  1.000000e-06
10  [[5.958742925531525], [14.327917968306403], [-…  1.000000e-02
4   [[9.970728371948313], [26.491694165903258], [-…  1.000000e-06
2   [[47.64433276868854], [130.6790054571249], [-1…  1.000000e-06
1   [[95.81933692733199], [262.7825656994754], [-3…  1.000000e-06
3   [[24.070810464619214], [65.46666343519061], [-…  1.000000e-06
9   [[0.0003065942121212121], [0.00038259484848484…  1.000000e-06
```

```
    elapsed_time is_max_reached        loss  accuracy_kfold       variable
16         0.413          False   16.097559        0.751515        epsilon
14         0.072          False   16.436353        0.751515        epsilon
6          0.105          False   16.436353        0.751515  learning rate
15         0.166          False   16.225842        0.751178        epsilon
17         0.940          False   16.019264        0.750842        epsilon
5          0.162          False   17.234261        0.750505  learning rate
13         0.027          False   16.775910        0.749495        epsilon
0          0.049          False   15.855793        0.749495   all features
7          0.155          False   15.849570        0.749495  learning rate
12         0.013          False   17.369442        0.747475        epsilon
11         0.009          False   18.607920        0.743771        epsilon
8          0.266          False   16.462365        0.741414  learning rate
10         0.009          False   21.366778        0.737037        epsilon
4          6.472           True   46.838322        0.688552  learning rate
2          6.500           True  106.221172        0.683502  learning rate
1          8.592           True  118.189617        0.683165  learning rate
3          8.383           True   87.970034        0.683165  learning rate
9          0.000          False   20.585959        0.516835  learning rate
```

[20]:
```
kd_x_train_np = pd.DataFrame(kd_x_train)
```

[21]:
```
#explore different logs of features
for i in range(kd_x_train.shape[1]):
    print (i,',log , column=>',kd_columns[i])
    kd_x_train_modified = log_transform_normalize(kd_x_train_np,i)
    title = 'log {}'.format(kd_columns[i])
    model_data = run_model('kd',title,learning_rate = 0, learning_rate_type =␣
↪learning_rate_type.iteration_plus_one ,max_iterations = 150000,
                          epsilon = 1e-6,x_train = kd_x_train_modified , y_train =␣
↪kd_y_train , model_data = model_data,variable = 'log')
show_sorted_model(model_data)
```

```
0 ,log , column=> Pregnancies
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
1 ,log , column=> Glucose
i======> 0
```

```
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
2 ,log , column=> BloodPressure
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
3 ,log , column=> Heart Rate
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
4 ,log , column=> SkinThickness
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
5 ,log , column=> Insulin
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
```

```
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
6 ,log , column=> BMI
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
7 ,log , column=> DiabetesPedigreeFunction
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
8 ,log , column=> Age
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[21]:

```
   model_name                                      description  \
23         kd                                      log Insulin
25         kd                     log DiabetesPedigreeFunction
26         kd                                          log Age
14         kd                        whole model-epsilon:1e-06
6          kd   whole model-lr:lr_type.iteration_plus_one
16         kd                        whole model-epsilon:1e-08
15         kd                        whole model-epsilon:1e-07
```

```
17         kd                   whole model-epsilon:1e-09
5          kd            whole model-lr:lr_type.iteration
18         kd                             log Pregnancies
0          kd                                 whole model
7          kd          whole model-lr:lr_type.sample_size
13         kd                   whole model-epsilon:1e-05
19         kd                                 log Glucose
12         kd                  whole model-epsilon:0.0001
24         kd                                     log BMI
22         kd                           log SkinThickness
21         kd                              log Heart Rate
11         kd                   whole model-epsilon:0.001
20         kd                           log BloodPressure
8          kd      whole model-lr:lr_type.ten_sample_size
10         kd                    whole model-epsilon:0.01
4          kd                         whole model-lr:0.05
2          kd                         whole model-lr:0.25
1          kd                          whole model-lr:0.5
3          kd                        whole model-lr:0.125
9          kd  whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
23   lr_type.iteration_plus_one       802
25   lr_type.iteration_plus_one      1118
26   lr_type.iteration_plus_one      1425
14   lr_type.iteration_plus_one      1476
6    lr_type.iteration_plus_one      1476
16   lr_type.iteration_plus_one      9021
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
5             lr_type.iteration      2220
18   lr_type.iteration_plus_one      1129
0                          0.01       956
7           lr_type.sample_size      2048
13   lr_type.iteration_plus_one       636
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
24   lr_type.iteration_plus_one      1341
22   lr_type.iteration_plus_one      1075
21   lr_type.iteration_plus_one      1008
11   lr_type.iteration_plus_one       165
20   lr_type.iteration_plus_one      1045
8       lr_type.ten_sample_size      3516
10   lr_type.iteration_plus_one        84
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
```

3		0.125	150000
9	lr_type.hundred_sample_size	2	

	weights	epsilon \
23	[[2.686500388124719], [8.005602930239544], [-0…	1.000000e-06
25	[[3.1317646256736045], [8.935591338567065], [-…	1.000000e-06
26	[[3.2317379647520714], [9.47235968785085], [-1…	1.000000e-06
14	[[3.175241989157885], [9.517996847051375], [-1…	1.000000e-06
6	[[3.175241989157885], [9.517996847051375], [-1…	1.000000e-06
16	[[2.96533714092659], [8.33217514426845], [-1.0…	1.000000e-08
15	[[3.0601381991207917], [8.83862515342757], [-1…	1.000000e-07
17	[[2.887385852612343], [7.9553323804450375], [-…	1.000000e-09
5	[[3.6602142620593963], [11.56162794637247], [-…	1.000000e-06
18	[[0.47048312539781034], [8.606990965621858], […	1.000000e-06
0	[[2.5819277747484066], [6.780007402086465], [-…	1.000000e-06
7	[[2.5011580589018196], [6.514304926750582], [-…	1.000000e-06
13	[[3.3327577160627104], [10.39377495841507], [-…	1.000000e-05
19	[[2.7341191226158608], [1.1070301791999142], […	1.000000e-06
12	[[3.6109084400585387], [11.42702526345321], [-…	1.000000e-04
24	[[3.016669053499772], [9.333632805388874], [-1…	1.000000e-06
22	[[3.097120568287582], [8.175406022439612], [-1…	1.000000e-06
21	[[2.7030305464057154], [8.683157932519787], [-…	1.000000e-06
11	[[4.254111597053624], [12.705390512862264], [-…	1.000000e-03
20	[[3.0167390363517588], [8.607754261039704], [0…	1.000000e-06
8	[[1.8917011563673924], [3.7425174840678586], […	1.000000e-06
10	[[5.958742925531525], [14.327917968306403], [-…	1.000000e-02
4	[[9.970728371948313], [26.491694165903258], [-…	1.000000e-06
2	[[47.64433276868854], [130.6790054571249], [-1…	1.000000e-06
1	[[95.81933692733199], [262.7825656994754], [-3…	1.000000e-06
3	[[24.070810464619214], [65.46666343519061], [-…	1.000000e-06
9	[[0.0003065942121212121], [0.00038259484848484…	1.000000e-06

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
23	0.037	False	15.821280	0.755556	log
25	0.051	False	16.145685	0.751852	log
26	0.070	False	16.404716	0.751515	log
14	0.072	False	16.436353	0.751515	epsilon
6	0.105	False	16.436353	0.751515	learning rate
16	0.413	False	16.097559	0.751515	epsilon
15	0.166	False	16.225842	0.751178	epsilon
17	0.940	False	16.019264	0.750842	epsilon
5	0.162	False	17.234261	0.750505	learning rate
18	0.048	False	16.526437	0.750168	log
0	0.049	False	15.855793	0.749495	all features
7	0.155	False	15.849570	0.749495	learning rate
13	0.027	False	16.775910	0.749495	epsilon
19	0.023	False	16.070200	0.748485	log

```
12        0.013      False   17.369442       0.747475        epsilon
24        0.056      False   16.234284       0.747138            log
22        0.069      False   16.334522       0.746465            log
21        0.044      False   16.245876       0.744781            log
11        0.009      False   18.607920       0.743771        epsilon
20        0.052      False   16.291746       0.743434            log
8         0.266      False   16.462365       0.741414  learning rate
10        0.009      False   21.366778       0.737037        epsilon
4         6.472       True   46.838322       0.688552  learning rate
2         6.500       True  106.221172       0.683502  learning rate
1         8.592       True  118.189617       0.683165  learning rate
3         8.383       True   87.970034       0.683165  learning rate
9         0.000      False   20.585959       0.516835  learning rate
```

[22]:

```
#combine log insulin-age
kd_x_train_log_insulin = log_transform_normalize(kd_x_train_np,5)
kd_x_train_logai = log_transform_normalize(kd_x_train_log_insulin,8)

model_data = run_model('kd','log age-insulin',learning_rate = 0,␣
 ↪learning_rate_type = learning_rate_type.iteration_plus_one ,max_iterations =␣
 ↪150000,
                       epsilon = 1e-6,x_train = kd_x_train_logai , y_train =␣
 ↪kd_y_train , model_data = model_data,variable = 'log')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[22]:

```
   model_name                                  description  \
27         kd                              log age-insulin
23         kd                                  log Insulin
25         kd                 log DiabetesPedigreeFunction
6          kd    whole model-lr:lr_type.iteration_plus_one
26         kd                                      log Age
16         kd                      whole model-epsilon:1e-08
14         kd                      whole model-epsilon:1e-06
15         kd                      whole model-epsilon:1e-07
17         kd                      whole model-epsilon:1e-09
5          kd              whole model-lr:lr_type.iteration
```

```
18          kd                              log Pregnancies
13          kd                    whole model-epsilon:1e-05
7           kd          whole model-lr:lr_type.sample_size
0           kd                                  whole model
19          kd                                  log Glucose
12          kd                   whole model-epsilon:0.0001
24          kd                                      log BMI
22          kd                            log SkinThickness
21          kd                               log Heart Rate
11          kd                    whole model-epsilon:0.001
20          kd                            log BloodPressure
8           kd      whole model-lr:lr_type.ten_sample_size
10          kd                     whole model-epsilon:0.01
4           kd                         whole model-lr:0.05
2           kd                         whole model-lr:0.25
1           kd                          whole model-lr:0.5
3           kd                        whole model-lr:0.125
9           kd  whole model-lr:lr_type.hundred_sample_size

                  learning_rate iteration  \
27   lr_type.iteration_plus_one       836
23   lr_type.iteration_plus_one       802
25   lr_type.iteration_plus_one      1118
6    lr_type.iteration_plus_one      1476
26   lr_type.iteration_plus_one      1425
16   lr_type.iteration_plus_one      9021
14   lr_type.iteration_plus_one      1476
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
5             lr_type.iteration      2220
18   lr_type.iteration_plus_one      1129
13   lr_type.iteration_plus_one       636
7           lr_type.sample_size      2048
0                          0.01       956
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
24   lr_type.iteration_plus_one      1341
22   lr_type.iteration_plus_one      1075
21   lr_type.iteration_plus_one      1008
11   lr_type.iteration_plus_one       165
20   lr_type.iteration_plus_one      1045
8       lr_type.ten_sample_size      3516
10   lr_type.iteration_plus_one        84
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
3                         0.125    150000
```

```
9   lr_type.hundred_sample_size         2

                                               weights       epsilon  \
27  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
23  [[2.686500388124719], [8.005602930239544], [-0…  1.000000e-06
25  [[3.1317646256736045], [8.935591338567065], [-…  1.000000e-06
6   [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
26  [[3.2317379647520714], [9.47235968785085], [-1…  1.000000e-06
16  [[2.96533714092659], [8.33217514426845], [-1.0…  1.000000e-08
14  [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
15  [[3.0601381991207917], [8.83862515342757], [-1…  1.000000e-07
17  [[2.887385852612343], [7.9553323804450375], [-…  1.000000e-09
5   [[3.6602142620593963], [11.56162794637247], [-…  1.000000e-06
18  [[0.47048312539781034], [8.606990965621858], […  1.000000e-06
13  [[3.3327577160627104], [10.39377495841507], [-…  1.000000e-05
7   [[2.5011580589018196], [6.514304926750582], [-…  1.000000e-06
0   [[2.5819277747484066], [6.780007402086465], [-…  1.000000e-06
19  [[2.7341191226158608], [1.1070301791999142], […  1.000000e-06
12  [[3.6109084400585387], [11.42702526345321], [-…  1.000000e-04
24  [[3.016669053499772], [9.333632805388874], [-1…  1.000000e-06
22  [[3.097120568287582], [8.175406022439612], [-1…  1.000000e-06
21  [[2.7030305464057154], [8.683157932519787], [-…  1.000000e-06
11  [[4.254111597053624], [12.705390512862264], [-…  1.000000e-03
20  [[3.0167390363517588], [8.607754261039704], [0…  1.000000e-06
8   [[1.8917011563673924], [3.7425174840678586], […  1.000000e-06
10  [[5.958742925531525], [14.327917968306403], [-…  1.000000e-02
4   [[9.970728371948313], [26.491694165903258], [-…  1.000000e-06
2   [[47.64433276868854], [130.6790054571249], [-1…  1.000000e-06
1   [[95.81933692733199], [262.7825656994754], [-3…  1.000000e-06
3   [[24.070810464619214], [65.46666343519061], [-…  1.000000e-06
9   [[0.0003065942121212121], [0.00038259484848484…  1.000000e-06

    elapsed_time is_max_reached       loss  accuracy_kfold       variable
27         0.035          False  15.737679        0.757576            log
23         0.037          False  15.821280        0.755556            log
25         0.051          False  16.145685        0.751852            log
6          0.105          False  16.436353        0.751515  learning rate
26         0.070          False  16.404716        0.751515            log
16         0.413          False  16.097559        0.751515        epsilon
14         0.072          False  16.436353        0.751515        epsilon
15         0.166          False  16.225842        0.751178        epsilon
17         0.940          False  16.019264        0.750842        epsilon
5          0.162          False  17.234261        0.750505  learning rate
18         0.048          False  16.526437        0.750168            log
13         0.027          False  16.775910        0.749495        epsilon
7          0.155          False  15.849570        0.749495  learning rate
0          0.049          False  15.855793        0.749495   all features
```

```
19          0.023           False    16.070200         0.748485            log
12          0.013           False    17.369442         0.747475        epsilon
24          0.056           False    16.234284         0.747138            log
22          0.069           False    16.334522         0.746465            log
21          0.044           False    16.245876         0.744781            log
11          0.009           False    18.607920         0.743771        epsilon
20          0.052           False    16.291746         0.743434            log
8           0.266           False    16.462365         0.741414  learning rate
10          0.009           False    21.366778         0.737037        epsilon
4           6.472            True    46.838322         0.688552  learning rate
2           6.500            True   106.221172         0.683502  learning rate
1           8.592            True   118.189617         0.683165  learning rate
3           8.383            True    87.970034         0.683165  learning rate
9           0.000           False    20.585959         0.516835  learning rate
```

[23]:

```
#combine log DiabetesPedigreeFunction
kd_x_train_logaid = log_transform_normalize(kd_x_train_logai,7)

model_data = run_model('kd','log age-insulin-Dpf',learning_rate = 0,␣
 ↪learning_rate_type = learning_rate_type.iteration_plus_one ,max_iterations =␣
 ↪150000,
                       epsilon = 1e-6,x_train = kd_x_train_logaid , y_train =␣
 ↪kd_y_train , model_data = model_data,variable = 'log')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[23]:

```
   model_name                                   description  \
27         kd                               log age-insulin
28         kd                           log age-insulin-Dpf
23         kd                                    log Insulin
25         kd                     log DiabetesPedigreeFunction
16         kd                       whole model-epsilon:1e-08
26         kd                                        log Age
14         kd                       whole model-epsilon:1e-06
6          kd   whole model-lr:lr_type.iteration_plus_one
15         kd                       whole model-epsilon:1e-07
17         kd                       whole model-epsilon:1e-09
```

```
5           kd            whole model-lr:lr_type.iteration
18          kd                             log Pregnancies
13          kd                  whole model-epsilon:1e-05
7           kd          whole model-lr:lr_type.sample_size
0           kd                                 whole model
19          kd                                 log Glucose
12          kd                whole model-epsilon:0.0001
24          kd                                     log BMI
22          kd                           log SkinThickness
21          kd                              log Heart Rate
11          kd                  whole model-epsilon:0.001
20          kd                           log BloodPressure
8           kd      whole model-lr:lr_type.ten_sample_size
10          kd                   whole model-epsilon:0.01
4           kd                        whole model-lr:0.05
2           kd                        whole model-lr:0.25
1           kd                         whole model-lr:0.5
3           kd                       whole model-lr:0.125
9           kd  whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
27   lr_type.iteration_plus_one       836
28   lr_type.iteration_plus_one       742
23   lr_type.iteration_plus_one       802
25   lr_type.iteration_plus_one      1118
16   lr_type.iteration_plus_one      9021
26   lr_type.iteration_plus_one      1425
14   lr_type.iteration_plus_one      1476
6    lr_type.iteration_plus_one      1476
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
5             lr_type.iteration      2220
18   lr_type.iteration_plus_one      1129
13   lr_type.iteration_plus_one       636
7           lr_type.sample_size      2048
0                          0.01       956
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
24   lr_type.iteration_plus_one      1341
22   lr_type.iteration_plus_one      1075
21   lr_type.iteration_plus_one      1008
11   lr_type.iteration_plus_one       165
20   lr_type.iteration_plus_one      1045
8       lr_type.ten_sample_size      3516
10   lr_type.iteration_plus_one        84
4                          0.05    150000
2                          0.25    150000
```

```
1                        0.5    150000
3                      0.125    150000
9   lr_type.hundred_sample_size      2

                                              weights       epsilon  \
27  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
28  [[2.81722351586145], [8.11597903143088], [-0.8…  1.000000e-06
23  [[2.686500388124719], [8.005602930239544], [-0…  1.000000e-06
25  [[3.1317646256736045], [8.935591338567065], [-…  1.000000e-06
16  [[2.96533714092659], [8.33217514426845], [-1.0…  1.000000e-08
26  [[3.2317379647520714], [9.47235968785085], [-1…  1.000000e-06
14  [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
6   [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
15  [[3.0601381991207917], [8.83862515342757], [-1…  1.000000e-07
17  [[2.887385852612343], [7.9553323804450375], [-…  1.000000e-09
5   [[3.6602142620593963], [11.56162794637247], [-…  1.000000e-06
18  [[0.47048312539781034], [8.606990965621858], […  1.000000e-06
13  [[3.3327577160627104], [10.39377495841507], [-…  1.000000e-05
7   [[2.5011580589018196], [6.514304926750582], [-…  1.000000e-06
0   [[2.5819277747484066], [6.780007402086465], [-…  1.000000e-06
19  [[2.7341191226158608], [1.1070301791999142], […  1.000000e-06
12  [[3.6109084400585387], [11.42702526345321], [-…  1.000000e-04
24  [[3.016669053499772], [9.333632805388874], [-1…  1.000000e-06
22  [[3.097120568287582], [8.175406022439612], [-1…  1.000000e-06
21  [[2.7030305464057154], [8.683157932519787], [-…  1.000000e-06
11  [[4.254111597053624], [12.705390512862264], [-…  1.000000e-03
20  [[3.0167390363517588], [8.607754261039704], [0…  1.000000e-06
8   [[1.8917011563673924], [3.7425174840678586], […  1.000000e-06
10  [[5.958742925531525], [14.327917968306403], [-…  1.000000e-02
4   [[9.970728371948313], [26.491694165903258], [-…  1.000000e-06
2   [[47.64433276868854], [130.6790054571249], [-1…  1.000000e-06
1   [[95.81933692733199], [262.7825656994754], [-3…  1.000000e-06
3   [[24.070810464619214], [65.46666343519061], [-…  1.000000e-06
9   [[0.0003065942121212121], [0.00038259484848484…  1.000000e-06

    elapsed_time is_max_reached       loss  accuracy_kfold       variable
27         0.035          False  15.737679        0.757576            log
28         0.036          False  15.629731        0.756902            log
23         0.037          False  15.821280        0.755556            log
25         0.051          False  16.145685        0.751852            log
16         0.413          False  16.097559        0.751515        epsilon
26         0.070          False  16.404716        0.751515            log
14         0.072          False  16.436353        0.751515        epsilon
6          0.105          False  16.436353        0.751515  learning rate
15         0.166          False  16.225842        0.751178        epsilon
17         0.940          False  16.019264        0.750842        epsilon
5          0.162          False  17.234261        0.750505  learning rate
```

```
18          0.048          False    16.526437         0.750168            log
13          0.027          False    16.775910         0.749495        epsilon
7           0.155          False    15.849570         0.749495  learning rate
0           0.049          False    15.855793         0.749495   all features
19          0.023          False    16.070200         0.748485            log
12          0.013          False    17.369442         0.747475        epsilon
24          0.056          False    16.234284         0.747138            log
22          0.069          False    16.334522         0.746465            log
21          0.044          False    16.245876         0.744781            log
11          0.009          False    18.607920         0.743771        epsilon
20          0.052          False    16.291746         0.743434            log
8           0.266          False    16.462365         0.741414  learning rate
10          0.009          False    21.366778         0.737037        epsilon
4           6.472           True    46.838322         0.688552  learning rate
2           6.500           True   106.221172         0.683502  learning rate
1           8.592           True   118.189617         0.683165  learning rate
3           8.383           True    87.970034         0.683165  learning rate
9           0.000          False    20.585959         0.516835  learning rate
```

[24]:
```
#combine log bmi-skintickness
kd_x_train_log_skin = log_transform_normalize(kd_x_train_np,4)
kd_x_train_skinbmi = log_transform_normalize(kd_x_train_log_skin,6)

model_data = run_model('kd','log skin-bmi',learning_rate = 0,␣
 ↪learning_rate_type = learning_rate_type.iteration_plus_one ,max_iterations =␣
 ↪150000,
                       epsilon = 1e-6,x_train = kd_x_train_skinbmi , y_train =␣
 ↪kd_y_train , model_data = model_data,variable = 'log')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[24]:
```
   model_name                              description  \
27         kd                          log age-insulin
28         kd                      log age-insulin-Dpf
23         kd                              log Insulin
25         kd               log DiabetesPedigreeFunction
26         kd                                  log Age
```

```
6          kd   whole model-lr:lr_type.iteration_plus_one
16         kd                  whole model-epsilon:1e-08
14         kd                  whole model-epsilon:1e-06
15         kd                  whole model-epsilon:1e-07
17         kd                  whole model-epsilon:1e-09
5          kd           whole model-lr:lr_type.iteration
18         kd                            log Pregnancies
13         kd                  whole model-epsilon:1e-05
0          kd                                whole model
7          kd         whole model-lr:lr_type.sample_size
19         kd                                log Glucose
12         kd                 whole model-epsilon:0.0001
24         kd                                    log BMI
22         kd                          log SkinThickness
21         kd                             log Heart Rate
29         kd                               log skin-bmi
11         kd                  whole model-epsilon:0.001
20         kd                          log BloodPressure
8          kd     whole model-lr:lr_type.ten_sample_size
10         kd                   whole model-epsilon:0.01
4          kd                       whole model-lr:0.05
2          kd                       whole model-lr:0.25
1          kd                        whole model-lr:0.5
3          kd                      whole model-lr:0.125
9          kd  whole model-lr:lr_type.hundred_sample_size

               learning_rate iteration  \
27   lr_type.iteration_plus_one       836
28   lr_type.iteration_plus_one       742
23   lr_type.iteration_plus_one       802
25   lr_type.iteration_plus_one      1118
26   lr_type.iteration_plus_one      1425
6    lr_type.iteration_plus_one      1476
16   lr_type.iteration_plus_one      9021
14   lr_type.iteration_plus_one      1476
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
5             lr_type.iteration      2220
18   lr_type.iteration_plus_one      1129
13   lr_type.iteration_plus_one       636
0                          0.01       956
7           lr_type.sample_size      2048
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
24   lr_type.iteration_plus_one      1341
22   lr_type.iteration_plus_one      1075
21   lr_type.iteration_plus_one      1008
```

```
29   lr_type.iteration_plus_one       978
11   lr_type.iteration_plus_one       165
20   lr_type.iteration_plus_one      1045
8       lr_type.ten_sample_size      3516
10   lr_type.iteration_plus_one        84
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
3                         0.125    150000
9   lr_type.hundred_sample_size         2

                                              weights       epsilon  \
27  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
28  [[2.81722351586145], [8.11597903143088], [-0.8…  1.000000e-06
23  [[2.686500388124719], [8.005602930239544], [-0…  1.000000e-06
25  [[3.1317646256736045], [8.935591338567065], [-…  1.000000e-06
26  [[3.2317379647520714], [9.47235968785085], [-1…  1.000000e-06
6   [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
16  [[2.96533714092659], [8.33217514426845], [-1.0…  1.000000e-08
14  [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
15  [[3.0601381991207917], [8.83862515342757], [-1…  1.000000e-07
17  [[2.887385852612343], [7.9553323804450375], [-…  1.000000e-09
5   [[3.6602142620593963], [11.56162794637247], [-…  1.000000e-06
18  [[0.47048312539781034], [8.606990965621858], […  1.000000e-06
13  [[3.3327577160627104], [10.39377495841507], [-…  1.000000e-05
0   [[2.5819277747484066], [6.780007402086465], [-…  1.000000e-06
7   [[2.5011580589018196], [6.514304926750582], [-…  1.000000e-06
19  [[2.7341191226158608], [1.1070301791999142], […  1.000000e-06
12  [[3.6109084400585387], [11.42702526345321], [-…  1.000000e-04
24  [[3.016669053499772], [9.333632805388874], [-1…  1.000000e-06
22  [[3.097120568287582], [8.175406022439612], [-1…  1.000000e-06
21  [[2.7030305464057154], [8.683157932519787], [-…  1.000000e-06
29  [[2.989559318463753], [8.044352212204572], [-1…  1.000000e-06
11  [[4.254111597053624], [12.705390512862264], [-…  1.000000e-03
20  [[3.0167390363517588], [8.607754261039704], [0…  1.000000e-06
8   [[1.8917011563673924], [3.7425174840678586], […  1.000000e-06
10  [[5.958742925531525], [14.327917968306403], [-…  1.000000e-02
4   [[9.970728371948313], [26.491694165903258], [-…  1.000000e-06
2   [[47.64433276868854], [130.6790054571249], [-1…  1.000000e-06
1   [[95.81933692733199], [262.7825656994754], [-3…  1.000000e-06
3   [[24.070810464619214], [65.46666343519061], [-…  1.000000e-06
9   [[0.0003065942121212121], [0.00038259484848484…  1.000000e-06

    elapsed_time is_max_reached       loss  accuracy_kfold       variable
27         0.035          False  15.737679        0.757576            log
28         0.036          False  15.629731        0.756902            log
23         0.037          False  15.821280        0.755556            log
```

```
25         0.051        False    16.145685       0.751852            log
26         0.070        False    16.404716       0.751515            log
6          0.105        False    16.436353       0.751515  learning rate
16         0.413        False    16.097559       0.751515        epsilon
14         0.072        False    16.436353       0.751515        epsilon
15         0.166        False    16.225842       0.751178        epsilon
17         0.940        False    16.019264       0.750842        epsilon
5          0.162        False    17.234261       0.750505  learning rate
18         0.048        False    16.526437       0.750168            log
13         0.027        False    16.775910       0.749495        epsilon
0          0.049        False    15.855793       0.749495   all features
7          0.155        False    15.849570       0.749495  learning rate
19         0.023        False    16.070200       0.748485            log
12         0.013        False    17.369442       0.747475        epsilon
24         0.056        False    16.234284       0.747138            log
22         0.069        False    16.334522       0.746465            log
21         0.044        False    16.245876       0.744781            log
29         0.061        False    16.176293       0.744108            log
11         0.009        False    18.607920       0.743771        epsilon
20         0.052        False    16.291746       0.743434            log
8          0.266        False    16.462365       0.741414  learning rate
10         0.009        False    21.366778       0.737037        epsilon
4          6.472         True    46.838322       0.688552  learning rate
2          6.500         True   106.221172       0.683502  learning rate
1          8.592         True   118.189617       0.683165  learning rate
3          8.383         True    87.970034       0.683165  learning rate
9          0.000        False    20.585959       0.516835  learning rate
```

[25]:

```
model_data = run_model('kd','log age-insulin',learning_rate = 0,␣
↪learning_rate_type = learning_rate_type.iteration_plus_one ,max_iterations =␣
↪150000,
                        epsilon = 1e-6,x_train = kd_x_train_logai , y_train =␣
↪kd_y_train , model_data = model_data,variable = 'log')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

```
[25]:    model_name                                  description  \
30           kd                           log age-insulin
27           kd                           log age-insulin
28           kd                       log age-insulin-Dpf
23           kd                                log Insulin
25           kd            log DiabetesPedigreeFunction
14           kd               whole model-epsilon:1e-06
26           kd                                    log Age
6            kd  whole model-lr:lr_type.iteration_plus_one
16           kd               whole model-epsilon:1e-08
15           kd               whole model-epsilon:1e-07
17           kd               whole model-epsilon:1e-09
5            kd         whole model-lr:lr_type.iteration
18           kd                           log Pregnancies
0            kd                                whole model
7            kd       whole model-lr:lr_type.sample_size
13           kd               whole model-epsilon:1e-05
19           kd                                log Glucose
12           kd              whole model-epsilon:0.0001
24           kd                                    log BMI
22           kd                         log SkinThickness
21           kd                            log Heart Rate
29           kd                              log skin-bmi
11           kd               whole model-epsilon:0.001
20           kd                         log BloodPressure
8            kd   whole model-lr:lr_type.ten_sample_size
10           kd                whole model-epsilon:0.01
4            kd                     whole model-lr:0.05
2            kd                     whole model-lr:0.25
1            kd                      whole model-lr:0.5
3            kd                    whole model-lr:0.125
9            kd whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
30  lr_type.iteration_plus_one       836
27  lr_type.iteration_plus_one       836
28  lr_type.iteration_plus_one       742
23  lr_type.iteration_plus_one       802
25  lr_type.iteration_plus_one      1118
14  lr_type.iteration_plus_one      1476
26  lr_type.iteration_plus_one      1425
6   lr_type.iteration_plus_one      1476
16  lr_type.iteration_plus_one      9021
15  lr_type.iteration_plus_one      3643
17  lr_type.iteration_plus_one     22164
5            lr_type.iteration      2220
18  lr_type.iteration_plus_one      1129
```

```
0                          0.01       956
7           lr_type.sample_size      2048
13   lr_type.iteration_plus_one       636
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
24   lr_type.iteration_plus_one      1341
22   lr_type.iteration_plus_one      1075
21   lr_type.iteration_plus_one      1008
29   lr_type.iteration_plus_one       978
11   lr_type.iteration_plus_one       165
20   lr_type.iteration_plus_one      1045
8       lr_type.ten_sample_size      3516
10   lr_type.iteration_plus_one        84
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
3                         0.125    150000
9   lr_type.hundred_sample_size         2

                                              weights       epsilon  \
30  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
27  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
28  [[2.81722351586145], [8.11597903143088], [-0.8…  1.000000e-06
23  [[2.686500388124719], [8.005602930239544], [-0…  1.000000e-06
25  [[3.1317646256736045], [8.935591338567065], [-…  1.000000e-06
14  [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
26  [[3.2317379647520714], [9.47235968785085], [-1…  1.000000e-06
6   [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
16  [[2.96533714092659], [8.33217514426845], [-1.0…  1.000000e-08
15  [[3.0601381991207917], [8.83862515342757], [-1…  1.000000e-07
17  [[2.887385852612343], [7.9553323804450375], [-…  1.000000e-09
5   [[3.6602142620593963], [11.56162794637247], [-…  1.000000e-06
18  [[0.47048312539781034], [8.606990965621858], […  1.000000e-06
0   [[2.5819277747484066], [6.780007402086465], [-…  1.000000e-06
7   [[2.5011580589018196], [6.514304926750582], [-…  1.000000e-06
13  [[3.3327577160627104], [10.39377495841507], [-…  1.000000e-05
19  [[2.7341191226158608], [1.1070301791999142], […  1.000000e-06
12  [[3.6109084400585387], [11.42702526345321], [-…  1.000000e-04
24  [[3.016669053499772], [9.333632805388874], [-1…  1.000000e-06
22  [[3.097120568287582], [8.175406022439612], [-1…  1.000000e-06
21  [[2.7030305464057154], [8.683157932519787], [-…  1.000000e-06
29  [[2.989559318463753], [8.044352212204572], [-1…  1.000000e-06
11  [[4.254111597053624], [12.705390512862264], [-…  1.000000e-03
20  [[3.0167390363517588], [8.607754261039704], [0…  1.000000e-06
8   [[1.8917011563673924], [3.7425174840678586], […  1.000000e-06
10  [[5.958742925531525], [14.327917968306403], [-…  1.000000e-02
4   [[9.970728371948313], [26.491694165903258], [-…  1.000000e-06
```

```
2    [[47.64433276868854], [130.6790054571249], [-1…  1.000000e-06
1    [[95.81933692733199], [262.7825656994754], [-3…  1.000000e-06
3    [[24.070810464619214], [65.46666343519061], [-…  1.000000e-06
9    [[0.0003065942121212121], [0.00038259484848484…  1.000000e-06

    elapsed_time  is_max_reached        loss  accuracy_kfold       variable
30         0.049           False   15.737679        0.757576            log
27         0.035           False   15.737679        0.757576            log
28         0.036           False   15.629731        0.756902            log
23         0.037           False   15.821280        0.755556            log
25         0.051           False   16.145685        0.751852            log
14         0.072           False   16.436353        0.751515        epsilon
26         0.070           False   16.404716        0.751515            log
6          0.105           False   16.436353        0.751515  learning rate
16         0.413           False   16.097559        0.751515        epsilon
15         0.166           False   16.225842        0.751178        epsilon
17         0.940           False   16.019264        0.750842        epsilon
5          0.162           False   17.234261        0.750505  learning rate
18         0.048           False   16.526437        0.750168            log
0          0.049           False   15.855793        0.749495   all features
7          0.155           False   15.849570        0.749495  learning rate
13         0.027           False   16.775910        0.749495        epsilon
19         0.023           False   16.070200        0.748485            log
12         0.013           False   17.369442        0.747475        epsilon
24         0.056           False   16.234284        0.747138            log
22         0.069           False   16.334522        0.746465            log
21         0.044           False   16.245876        0.744781            log
29         0.061           False   16.176293        0.744108            log
11         0.009           False   18.607920        0.743771        epsilon
20         0.052           False   16.291746        0.743434            log
8          0.266           False   16.462365        0.741414  learning rate
10         0.009           False   21.366778        0.737037        epsilon
4          6.472            True   46.838322        0.688552  learning rate
2          6.500            True  106.221172        0.683502  learning rate
1          8.592            True  118.189617        0.683165  learning rate
3          8.383            True   87.970034        0.683165  learning rate
9          0.000           False   20.585959        0.516835  learning rate
```

[26]:

```
#check for removing variables
for i in range(kd_x_train_logai.shape[1]):
    print (i,',column=>',kd_columns[i])
    kd_x_train_modified = np.delete(kd_x_train_logai.to_numpy(), [i], 1)
    title = 'log age-insulin-no {}'.format(kd_columns[i])
    model_data = run_model('kd',title,learning_rate = 0, learning_rate_type =␣
↪learning_rate_type.iteration_plus_one ,max_iterations = 150000,
                       epsilon = 1e-6,x_train = kd_x_train_modified , y_train =␣
↪kd_y_train , model_data = model_data,variable = 'feature removal')
```

```
show_sorted_model(model_data)
```

```
0 ,column=> Pregnancies
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
1 ,column=> Glucose
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
2 ,column=> BloodPressure
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
3 ,column=> Heart Rate
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
4 ,column=> SkinThickness
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
5 ,column=> Insulin
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
6 ,column=> BMI
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
7 ,column=> DiabetesPedigreeFunction
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
8 ,column=> Age
i======> 0
i======> 1
i======> 2
i======> 3
```

```
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[26]:

```
   model_name                                    description  \
34         kd                  log age-insulin-no Heart Rate
27         kd                                log age-insulin
30         kd                                log age-insulin
28         kd                            log age-insulin-Dpf
23         kd                                    log Insulin
38         kd  log age-insulin-no DiabetesPedigreeFunction
25         kd                 log DiabetesPedigreeFunction
16         kd                  whole model-epsilon:1e-08
26         kd                                        log Age
14         kd                  whole model-epsilon:1e-06
6          kd    whole model-lr:lr_type.iteration_plus_one
15         kd                  whole model-epsilon:1e-07
17         kd                  whole model-epsilon:1e-09
31         kd                 log age-insulin-no Pregnancies
33         kd               log age-insulin-no BloodPressure
35         kd               log age-insulin-no SkinThickness
5          kd               whole model-lr:lr_type.iteration
18         kd                                log Pregnancies
7          kd             whole model-lr:lr_type.sample_size
0          kd                                    whole model
13         kd                  whole model-epsilon:1e-05
19         kd                                    log Glucose
12         kd                 whole model-epsilon:0.0001
24         kd                                        log BMI
22         kd                              log SkinThickness
37         kd                         log age-insulin-no BMI
21         kd                                 log Heart Rate
29         kd                                   log skin-bmi
11         kd                  whole model-epsilon:0.001
20         kd                              log BloodPressure
8          kd        whole model-lr:lr_type.ten_sample_size
39         kd                         log age-insulin-no Age
36         kd                     log age-insulin-no Insulin
10         kd                   whole model-epsilon:0.01
32         kd                     log age-insulin-no Glucose
4          kd                         whole model-lr:0.05
2          kd                         whole model-lr:0.25
1          kd                          whole model-lr:0.5
3          kd                        whole model-lr:0.125
```

```
9         kd   whole model-lr:lr_type.hundred_sample_size

                  learning_rate iteration  \
34   lr_type.iteration_plus_one       813
27   lr_type.iteration_plus_one       836
30   lr_type.iteration_plus_one       836
28   lr_type.iteration_plus_one       742
23   lr_type.iteration_plus_one       802
38   lr_type.iteration_plus_one       995
25   lr_type.iteration_plus_one      1118
16   lr_type.iteration_plus_one      9021
26   lr_type.iteration_plus_one      1425
14   lr_type.iteration_plus_one      1476
6    lr_type.iteration_plus_one      1476
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
31   lr_type.iteration_plus_one       877
33   lr_type.iteration_plus_one       656
35   lr_type.iteration_plus_one       987
5             lr_type.iteration      2220
18   lr_type.iteration_plus_one      1129
7           lr_type.sample_size      2048
0                          0.01       956
13   lr_type.iteration_plus_one       636
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
24   lr_type.iteration_plus_one      1341
22   lr_type.iteration_plus_one      1075
37   lr_type.iteration_plus_one       711
21   lr_type.iteration_plus_one      1008
29   lr_type.iteration_plus_one       978
11   lr_type.iteration_plus_one       165
20   lr_type.iteration_plus_one      1045
8       lr_type.ten_sample_size      3516
39   lr_type.iteration_plus_one       790
36   lr_type.iteration_plus_one      1293
10   lr_type.iteration_plus_one        84
32   lr_type.iteration_plus_one       484
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
3                         0.125    150000
9  lr_type.hundred_sample_size         2

                                              weights       epsilon  \
34  [[2.7129340471248065], [8.273684993858577], [-…  1.000000e-06
27  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
```

```
30   [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
28   [[2.81722351586145], [8.11597903143088], [-0.8…  1.000000e-06
23   [[2.686500388124719], [8.005602930239544], [-0…  1.000000e-06
38   [[3.528294663986213], [8.888662016489207], [-0…  1.000000e-06
25   [[3.1317646256736045], [8.935591338567065], [-…  1.000000e-06
16   [[2.96533714092659], [8.33217514426845], [-1.0…  1.000000e-08
26   [[3.2317379647520714], [9.47235968785085], [-1…  1.000000e-06
14   [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
6    [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
15   [[3.0601381991207917], [8.83862515342757], [-1…  1.000000e-07
17   [[2.887385852612343], [7.9553323804450375], [-…  1.000000e-09
31   [[7.954655592251466], [-1.0448422866347953], […  1.000000e-06
33   [[2.726699034298592], [7.741999408211606], [-0…  1.000000e-06
35   [[2.8454163866127455], [7.444158178181417], [-…  1.000000e-06
5    [[3.6602142620593963], [11.56162794637247], [-…  1.000000e-06
18   [[0.47048312539781034], [8.606990965621858], […  1.000000e-06
7    [[2.5011580589018196], [6.514304926750582], [-…  1.000000e-06
0    [[2.5819277747484066], [6.780007402086465], [-…  1.000000e-06
13   [[3.3327577160627104], [10.39377495841507], [-…  1.000000e-05
19   [[2.7341191226158608], [1.1070301791999142], […  1.000000e-06
12   [[3.6109084400585387], [11.42702526345321], [-…  1.000000e-04
24   [[3.016669053499772], [9.333632805388874], [-1…  1.000000e-06
22   [[3.097120568287582], [8.175406022439612], [-1…  1.000000e-06
37   [[2.5109910019024677], [7.816325231383992], [-…  1.000000e-06
21   [[2.7030305464057154], [8.683157932519787], [-…  1.000000e-06
29   [[2.989559318463753], [8.044352212204572], [-1…  1.000000e-06
11   [[4.254111597053624], [12.705390512862264], [-…  1.000000e-03
20   [[3.0167390363517588], [8.607754261039704], [0…  1.000000e-06
8    [[1.8917011563673924], [3.7425174840678586], […  1.000000e-06
39   [[2.5616285768893263], [7.794299182419026], [-…  1.000000e-06
36   [[2.820836202474606], [9.936728220495777], [-0…  1.000000e-06
10   [[5.958742925531525], [14.327917968306403], [-…  1.000000e-02
32   [[2.360917183337187], [-0.8182987407890269], […  1.000000e-06
4    [[9.970728371948313], [26.491694165903258], [-…  1.000000e-06
2    [[47.64433276868854], [130.6790054571249], [-1…  1.000000e-06
1    [[95.81933692733199], [262.7825656994754], [-3…  1.000000e-06
3    [[24.070810464619214], [65.46666343519061], [-…  1.000000e-06
9    [[0.0003065942121212121], [0.00038259484848484…  1.000000e-06

    elapsed_time is_max_reached       loss  accuracy_kfold         variable
34         0.062          False  15.717127        0.759933  feature removal
27         0.035          False  15.737679        0.757576              log
30         0.049          False  15.737679        0.757576              log
28         0.036          False  15.629731        0.756902              log
23         0.037          False  15.821280        0.755556              log
38         0.072          False  15.632706        0.751852  feature removal
25         0.051          False  16.145685        0.751852              log
```

16	0.413	False	16.097559	0.751515	epsilon
26	0.070	False	16.404716	0.751515	log
14	0.072	False	16.436353	0.751515	epsilon
6	0.105	False	16.436353	0.751515	learning rate
15	0.166	False	16.225842	0.751178	epsilon
17	0.940	False	16.019264	0.750842	epsilon
31	0.041	False	16.310581	0.750842	feature removal
33	0.046	False	15.583091	0.750842	feature removal
35	0.073	False	15.788618	0.750505	feature removal
5	0.162	False	17.234261	0.750505	learning rate
18	0.048	False	16.526437	0.750168	log
7	0.155	False	15.849570	0.749495	learning rate
0	0.049	False	15.855793	0.749495	all features
13	0.027	False	16.775910	0.749495	epsilon
19	0.023	False	16.070200	0.748485	log
12	0.013	False	17.369442	0.747475	epsilon
24	0.056	False	16.234284	0.747138	log
22	0.069	False	16.334522	0.746465	log
37	0.049	False	15.701077	0.746465	feature removal
21	0.044	False	16.245876	0.744781	log
29	0.061	False	16.176293	0.744108	log
11	0.009	False	18.607920	0.743771	epsilon
20	0.052	False	16.291746	0.743434	log
8	0.266	False	16.462365	0.741414	learning rate
39	0.061	False	15.972236	0.740067	feature removal
36	0.099	False	17.179665	0.739731	feature removal
10	0.009	False	21.366778	0.737037	epsilon
32	0.035	False	17.264986	0.729630	feature removal
4	6.472	True	46.838322	0.688552	learning rate
2	6.500	True	106.221172	0.683502	learning rate
1	8.592	True	118.189617	0.683165	learning rate
3	8.383	True	87.970034	0.683165	learning rate
9	0.000	False	20.585959	0.516835	learning rate

[27]:
```
#remove heartrate from features
kd_x_train_features = np.delete(kd_x_train_logai.to_numpy(), [3], 1)
title = 'log age-insulin-no heartrate'.format(kd_columns[i])
model_data = run_model('kd',title,learning_rate = 0, learning_rate_type =␣
↪learning_rate_type.iteration_plus_one ,max_iterations = 150000,
                    epsilon = 1e-6,x_train = kd_x_train_features , y_train =␣
↪kd_y_train , model_data = model_data,variable = 'feature removal')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
```

```
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[27]:

```
   model_name                                       description  \
40         kd                    log age-insulin-no heartrate
34         kd                   log age-insulin-no Heart Rate
27         kd                                 log age-insulin
30         kd                                 log age-insulin
28         kd                             log age-insulin-Dpf
23         kd                                     log Insulin
38         kd  log age-insulin-no DiabetesPedigreeFunction
25         kd                    log DiabetesPedigreeFunction
6          kd    whole model-lr:lr_type.iteration_plus_one
26         kd                                         log Age
14         kd                      whole model-epsilon:1e-06
16         kd                      whole model-epsilon:1e-08
15         kd                      whole model-epsilon:1e-07
17         kd                      whole model-epsilon:1e-09
33         kd                log age-insulin-no BloodPressure
31         kd                  log age-insulin-no Pregnancies
35         kd                log age-insulin-no SkinThickness
5          kd                whole model-lr:lr_type.iteration
18         kd                                 log Pregnancies
0          kd                                     whole model
13         kd                      whole model-epsilon:1e-05
7          kd              whole model-lr:lr_type.sample_size
19         kd                                     log Glucose
12         kd                     whole model-epsilon:0.0001
24         kd                                         log BMI
37         kd                          log age-insulin-no BMI
22         kd                               log SkinThickness
21         kd                                  log Heart Rate
29         kd                                    log skin-bmi
11         kd                      whole model-epsilon:0.001
20         kd                               log BloodPressure
8          kd          whole model-lr:lr_type.ten_sample_size
39         kd                          log age-insulin-no Age
36         kd                      log age-insulin-no Insulin
10         kd                       whole model-epsilon:0.01
32         kd                      log age-insulin-no Glucose
4          kd                             whole model-lr:0.05
2          kd                             whole model-lr:0.25
1          kd                              whole model-lr:0.5
```

```
3          kd                         whole model-lr:0.125
9          kd   whole model-lr:lr_type.hundred_sample_size

                  learning_rate iteration  \
40   lr_type.iteration_plus_one       813
34   lr_type.iteration_plus_one       813
27   lr_type.iteration_plus_one       836
30   lr_type.iteration_plus_one       836
28   lr_type.iteration_plus_one       742
23   lr_type.iteration_plus_one       802
38   lr_type.iteration_plus_one       995
25   lr_type.iteration_plus_one      1118
6    lr_type.iteration_plus_one      1476
26   lr_type.iteration_plus_one      1425
14   lr_type.iteration_plus_one      1476
16   lr_type.iteration_plus_one      9021
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
33   lr_type.iteration_plus_one       656
31   lr_type.iteration_plus_one       877
35   lr_type.iteration_plus_one       987
5             lr_type.iteration      2220
18   lr_type.iteration_plus_one      1129
0                          0.01       956
13   lr_type.iteration_plus_one       636
7           lr_type.sample_size      2048
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
24   lr_type.iteration_plus_one      1341
37   lr_type.iteration_plus_one       711
22   lr_type.iteration_plus_one      1075
21   lr_type.iteration_plus_one      1008
29   lr_type.iteration_plus_one       978
11   lr_type.iteration_plus_one       165
20   lr_type.iteration_plus_one      1045
8       lr_type.ten_sample_size      3516
39   lr_type.iteration_plus_one       790
36   lr_type.iteration_plus_one      1293
10   lr_type.iteration_plus_one        84
32   lr_type.iteration_plus_one       484
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
3                         0.125    150000
9  lr_type.hundred_sample_size          2

                                            weights       epsilon  \
```

```
40  [[2.7129340471248065], [8.273684993858577], [-…  1.000000e-06
34  [[2.7129340471248065], [8.273684993858577], [-…  1.000000e-06
27  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
30  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
28  [[2.81722351586145], [8.11597903143088], [-0.8…  1.000000e-06
23  [[2.686500388124719], [8.005602930239544], [-0…  1.000000e-06
38  [[3.528294663986213], [8.888662016489207], [-0…  1.000000e-06
25  [[3.1317646256736045], [8.935591338567065], [-…  1.000000e-06
6   [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
26  [[3.2317379647520714], [9.47235968785085], [-1…  1.000000e-06
14  [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
16  [[2.96533714092659], [8.33217514426845], [-1.0…  1.000000e-08
15  [[3.0601381991207917], [8.83862515342757], [-1…  1.000000e-07
17  [[2.887385852612343], [7.9553323804450375], [-…  1.000000e-09
33  [[2.726699034298592], [7.741999408211606], [-0…  1.000000e-06
31  [[7.954655592251466], [-1.0448422866347953], […  1.000000e-06
35  [[2.8454163866127455], [7.444158178181417], [-…  1.000000e-06
5   [[3.6602142620593963], [11.56162794637247], [-…  1.000000e-06
18  [[0.47048312539781034], [8.606990965621858], […  1.000000e-06
0   [[2.5819277747484066], [6.780007402086465], [-…  1.000000e-06
13  [[3.3327577160627104], [10.39377495841507], [-…  1.000000e-05
7   [[2.5011580589018196], [6.514304926750582], [-…  1.000000e-06
19  [[2.7341191226158608], [1.1070301791999142], […  1.000000e-06
12  [[3.6109084400585387], [11.42702526345321], [-…  1.000000e-04
24  [[3.016669053499772], [9.333632805388874], [-1…  1.000000e-06
37  [[2.5109910019024677], [7.816325231383992], [-…  1.000000e-06
22  [[3.097120568287582], [8.175406022439612], [-1…  1.000000e-06
21  [[2.7030305464057154], [8.683157932519787], [-…  1.000000e-06
29  [[2.989559318463753], [8.044352212204572], [-1…  1.000000e-06
11  [[4.254111597053624], [12.705390512862264], [-…  1.000000e-03
20  [[3.0167390363517588], [8.607754261039704], [0…  1.000000e-06
8   [[1.8917011563673924], [3.7425174840678586], […  1.000000e-06
39  [[2.5616285768893263], [7.794299182419026], [-…  1.000000e-06
36  [[2.820836202474606], [9.936728220495777], [-0…  1.000000e-06
10  [[5.958742925531525], [14.327917968306403], [-…  1.000000e-02
32  [[2.360917183337187], [-0.8182987407890269], […  1.000000e-06
4   [[9.970728371948313], [26.491694165903258], [-…  1.000000e-06
2   [[47.64433276868854], [130.6790054571249], [-1…  1.000000e-06
1   [[95.81933692733199], [262.7825656994754], [-3…  1.000000e-06
3   [[24.070810464619214], [65.46666343519061], [-…  1.000000e-06
9   [[0.0003065942121212121], [0.00038259484848484…  1.000000e-06

    elapsed_time is_max_reached       loss  accuracy_kfold         variable
40         0.045          False  15.717127        0.759933  feature removal
34         0.062          False  15.717127        0.759933  feature removal
27         0.035          False  15.737679        0.757576              log
30         0.049          False  15.737679        0.757576              log
```

28	0.036	False	15.629731	0.756902	log
23	0.037	False	15.821280	0.755556	log
38	0.072	False	15.632706	0.751852	feature removal
25	0.051	False	16.145685	0.751852	log
6	0.105	False	16.436353	0.751515	learning rate
26	0.070	False	16.404716	0.751515	log
14	0.072	False	16.436353	0.751515	epsilon
16	0.413	False	16.097559	0.751515	epsilon
15	0.166	False	16.225842	0.751178	epsilon
17	0.940	False	16.019264	0.750842	epsilon
33	0.046	False	15.583091	0.750842	feature removal
31	0.041	False	16.310581	0.750842	feature removal
35	0.073	False	15.788618	0.750505	feature removal
5	0.162	False	17.234261	0.750505	learning rate
18	0.048	False	16.526437	0.750168	log
0	0.049	False	15.855793	0.749495	all features
13	0.027	False	16.775910	0.749495	epsilon
7	0.155	False	15.849570	0.749495	learning rate
19	0.023	False	16.070200	0.748485	log
12	0.013	False	17.369442	0.747475	epsilon
24	0.056	False	16.234284	0.747138	log
37	0.049	False	15.701077	0.746465	feature removal
22	0.069	False	16.334522	0.746465	log
21	0.044	False	16.245876	0.744781	log
29	0.061	False	16.176293	0.744108	log
11	0.009	False	18.607920	0.743771	epsilon
20	0.052	False	16.291746	0.743434	log
8	0.266	False	16.462365	0.741414	learning rate
39	0.061	False	15.972236	0.740067	feature removal
36	0.099	False	17.179665	0.739731	feature removal
10	0.009	False	21.366778	0.737037	epsilon
32	0.035	False	17.264986	0.729630	feature removal
4	6.472	True	46.838322	0.688552	learning rate
2	6.500	True	106.221172	0.683502	learning rate
1	8.592	True	118.189617	0.683165	learning rate
3	8.383	True	87.970034	0.683165	learning rate
9	0.000	False	20.585959	0.516835	learning rate

[28]:
```
#check power 2 of features
for i in range(kd_x_train_features.shape[1]):
    print (i,',column=>',kd_columns[i])
    kd_x_power = power_n_feature(pd.DataFrame(kd_x_train_features),i,2).
 ↪to_numpy()
    title = 'log age-insulin-no heartrate-*2 {}'.format(kd_columns[i])
    print(title)
    model_data = run_model('kd',title,learning_rate = 0, learning_rate_type =␣
 ↪learning_rate_type.iteration_plus_one ,max_iterations = 150000,
```

```
                            epsilon = 1e-6,x_train = kd_x_power , y_train =␣
↪kd_y_train , model_data = model_data,variable = 'feature *2')
show_sorted_model(model_data)
```

```
0 ,column=> Pregnancies
log age-insulin-no heartrate-*2 Pregnancies
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
1 ,column=> Glucose
log age-insulin-no heartrate-*2 Glucose
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
2 ,column=> BloodPressure
log age-insulin-no heartrate-*2 BloodPressure
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
3 ,column=> Heart Rate
log age-insulin-no heartrate-*2 Heart Rate
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
```

```
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
4 ,column=> SkinThickness
log age-insulin-no heartrate-*2 SkinThickness
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
5 ,column=> Insulin
log age-insulin-no heartrate-*2 Insulin
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
6 ,column=> BMI
log age-insulin-no heartrate-*2 BMI
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
7 ,column=> DiabetesPedigreeFunction
log age-insulin-no heartrate-*2 DiabetesPedigreeFunction
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
```

```
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[28]:

```
   model_name                                        description  \
34         kd                      log age-insulin-no Heart Rate
40         kd                       log age-insulin-no heartrate
42         kd             log age-insulin-no heartrate-*2 Glucose
43         kd       log age-insulin-no heartrate-*2 BloodPressure
41         kd         log age-insulin-no heartrate-*2 Pregnancies
27         kd                                    log age-insulin
30         kd                                    log age-insulin
28         kd                                log age-insulin-Dpf
47         kd                 log age-insulin-no heartrate-*2 BMI
46         kd             log age-insulin-no heartrate-*2 Insulin
23         kd                                        log Insulin
38         kd         log age-insulin-no DiabetesPedigreeFunction
25         kd                       log DiabetesPedigreeFunction
6          kd           whole model-lr:lr_type.iteration_plus_one
26         kd                                            log Age
14         kd                         whole model-epsilon:1e-06
16         kd                         whole model-epsilon:1e-08
15         kd                         whole model-epsilon:1e-07
17         kd                         whole model-epsilon:1e-09
33         kd                   log age-insulin-no BloodPressure
31         kd                     log age-insulin-no Pregnancies
35         kd                   log age-insulin-no SkinThickness
5          kd                      whole model-lr:lr_type.iteration
44         kd          log age-insulin-no heartrate-*2 Heart Rate
18         kd                                    log Pregnancies
13         kd                         whole model-epsilon:1e-05
7          kd                    whole model-lr:lr_type.sample_size
0          kd                                        whole model
19         kd                                        log Glucose
12         kd                        whole model-epsilon:0.0001
24         kd                                            log BMI
37         kd                             log age-insulin-no BMI
22         kd                                  log SkinThickness
21         kd                                     log Heart Rate
29         kd                                       log skin-bmi
11         kd                         whole model-epsilon:0.001
48         kd  log age-insulin-no heartrate-*2 DiabetesPedigr…
20         kd                                  log BloodPressure
8          kd                whole model-lr:lr_type.ten_sample_size
39         kd                             log age-insulin-no Age
```

```
36        kd                      log age-insulin-no Insulin
10        kd                        whole model-epsilon:0.01
45        kd      log age-insulin-no heartrate-*2 SkinThickness
32        kd                      log age-insulin-no Glucose
4         kd                             whole model-lr:0.05
2         kd                             whole model-lr:0.25
1         kd                              whole model-lr:0.5
3         kd                            whole model-lr:0.125
9         kd         whole model-lr:lr_type.hundred_sample_size

                   learning_rate iteration  \
34   lr_type.iteration_plus_one       813
40   lr_type.iteration_plus_one       813
42   lr_type.iteration_plus_one       706
43   lr_type.iteration_plus_one       702
41   lr_type.iteration_plus_one       832
27   lr_type.iteration_plus_one       836
30   lr_type.iteration_plus_one       836
28   lr_type.iteration_plus_one       742
47   lr_type.iteration_plus_one       938
46   lr_type.iteration_plus_one       741
23   lr_type.iteration_plus_one       802
38   lr_type.iteration_plus_one       995
25   lr_type.iteration_plus_one      1118
6    lr_type.iteration_plus_one      1476
26   lr_type.iteration_plus_one      1425
14   lr_type.iteration_plus_one      1476
16   lr_type.iteration_plus_one      9021
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
33   lr_type.iteration_plus_one       656
31   lr_type.iteration_plus_one       877
35   lr_type.iteration_plus_one       987
5             lr_type.iteration      2220
44   lr_type.iteration_plus_one       834
18   lr_type.iteration_plus_one      1129
13   lr_type.iteration_plus_one       636
7           lr_type.sample_size      2048
0                          0.01       956
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
24   lr_type.iteration_plus_one      1341
37   lr_type.iteration_plus_one       711
22   lr_type.iteration_plus_one      1075
21   lr_type.iteration_plus_one      1008
29   lr_type.iteration_plus_one       978
11   lr_type.iteration_plus_one       165
```

```
48   lr_type.iteration_plus_one      1274
20   lr_type.iteration_plus_one      1045
8       lr_type.ten_sample_size      3516
39   lr_type.iteration_plus_one       790
36   lr_type.iteration_plus_one      1293
10   lr_type.iteration_plus_one        84
45   lr_type.iteration_plus_one      1507
32   lr_type.iteration_plus_one       484
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
3                         0.125    150000
9   lr_type.hundred_sample_size         2

                                                 weights       epsilon  \
34  [[2.7129340471248065], [8.273684993858577], [-…  1.000000e-06
40  [[2.7129340471248065], [8.273684993858577], [-…  1.000000e-06
42  [[2.7707684939995394], [5.782931625664704], [-…  1.000000e-06
43  [[2.6840445568619615], [8.100362853592744], [-…  1.000000e-06
41  [[3.7667195356425127], [8.238912283417783], [-…  1.000000e-06
27  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
30  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
28  [[2.81722351586145], [8.11597903143088], [-0.8…  1.000000e-06
47  [[3.1911630539778977], [8.668229576381282], [-…  1.000000e-06
46  [[2.5822636064767592], [7.952642108792551], [-…  1.000000e-06
23  [[2.686500388124719], [8.005602930239544], [-0…  1.000000e-06
38  [[3.528294663986213], [8.888662016489207], [-0…  1.000000e-06
25  [[3.1317646256736045], [8.935591338567065], [-…  1.000000e-06
6   [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
26  [[3.2317379647520714], [9.47235968785085], [-1…  1.000000e-06
14  [[3.175241989157885], [9.517996847051375], [-1…  1.000000e-06
16  [[2.96533714092659], [8.33217514426845], [-1.0…  1.000000e-08
15  [[3.0601381991207917], [8.83862515342757], [-1…  1.000000e-07
17  [[2.887385852612343], [7.9553323804450375], [-…  1.000000e-09
33  [[2.726699034298592], [7.741999408211606], [-0…  1.000000e-06
31  [[7.954655592251466], [-1.0448422866347953], […  1.000000e-06
35  [[2.8454163866127455], [7.444158178181417], [-…  1.000000e-06
5   [[3.6602142620593963], [11.56162794637247], [-…  1.000000e-06
44  [[2.724842549802258], [8.017986056469372], [-1…  1.000000e-06
18  [[0.47048312539781034], [8.606990965621858], […  1.000000e-06
13  [[3.3327577160627104], [10.39377495841507], [-…  1.000000e-05
7   [[2.5011580589018196], [6.514304926750582], [-…  1.000000e-06
0   [[2.5819277747484066], [6.780007402086465], [-…  1.000000e-06
19  [[2.7341191226158608], [1.1070301791999142], […  1.000000e-06
12  [[3.6109084400585387], [11.42702526345321], [-…  1.000000e-04
24  [[3.016669053499772], [9.333632805388874], [-1…  1.000000e-06
37  [[2.510991001902467 7], [7.816325231383992], [-…  1.000000e-06
```

22	[[3.097120568287582], [8.175406022439612], [-1…	1.000000e-06
21	[[2.7030305464057154], [8.683157932519787], [-…	1.000000e-06
29	[[2.989559318463753], [8.044352212204572], [-1…	1.000000e-06
11	[[4.254111597053624], [12.705390512862264], [-…	1.000000e-03
48	[[2.8730033537390356], [8.894450845054584], [-…	1.000000e-06
20	[[3.0167390363517588], [8.607754261039704], [0…	1.000000e-06
8	[[1.8917011563673924], [3.7425174840678586], […	1.000000e-06
39	[[2.5616285768893263], [7.794299182419026], [-…	1.000000e-06
36	[[2.820836202474606], [9.936728220495777], [-0…	1.000000e-06
10	[[5.958742925531525], [14.327917968306403], [-…	1.000000e-02
45	[[2.8269443408264547], [10.095837895438326], […	1.000000e-06
32	[[2.360917183337187], [-0.8182987407890269], […	1.000000e-06
4	[[9.970728371948313], [26.491694165903258], [-…	1.000000e-06
2	[[47.64433276868854], [130.6790054571249], [-1…	1.000000e-06
1	[[95.81933692733199], [262.7825656994754], [-3…	1.000000e-06
3	[[24.070810464619214], [65.46666343519061], [-…	1.000000e-06
9	[[0.0003065942121212121], [0.00038259484848484…	1.000000e-06

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
34	0.062	False	15.717127	0.759933	feature removal
40	0.045	False	15.717127	0.759933	feature removal
42	0.032	False	15.705729	0.758586	feature *2
43	0.033	False	15.651793	0.758249	feature *2
41	0.046	False	15.915845	0.758249	feature *2
27	0.035	False	15.737679	0.757576	log
30	0.049	False	15.737679	0.757576	log
28	0.036	False	15.629731	0.756902	log
47	0.039	False	15.849282	0.756566	feature *2
46	0.031	False	15.840965	0.755556	feature *2
23	0.037	False	15.821280	0.755556	log
38	0.072	False	15.632706	0.751852	feature removal
25	0.051	False	16.145685	0.751852	log
6	0.105	False	16.436353	0.751515	learning rate
26	0.070	False	16.404716	0.751515	log
14	0.072	False	16.436353	0.751515	epsilon
16	0.413	False	16.097559	0.751515	epsilon
15	0.166	False	16.225842	0.751178	epsilon
17	0.940	False	16.019264	0.750842	epsilon
33	0.046	False	15.583091	0.750842	feature removal
31	0.041	False	16.310581	0.750842	feature removal
35	0.073	False	15.788618	0.750505	feature removal
5	0.162	False	17.234261	0.750505	learning rate
44	0.036	False	15.823767	0.750168	feature *2
18	0.048	False	16.526437	0.750168	log
13	0.027	False	16.775910	0.749495	epsilon
7	0.155	False	15.849570	0.749495	learning rate
0	0.049	False	15.855793	0.749495	all features

```
19        0.023     False    16.070200     0.748485              log
12        0.013     False    17.369442     0.747475          epsilon
24        0.056     False    16.234284     0.747138              log
37        0.049     False    15.701077     0.746465  feature removal
22        0.069     False    16.334522     0.746465              log
21        0.044     False    16.245876     0.744781              log
29        0.061     False    16.176293     0.744108              log
11        0.009     False    18.607920     0.743771          epsilon
48        0.055     False    16.098061     0.743771       feature *2
20        0.052     False    16.291746     0.743434              log
8         0.266     False    16.462365     0.741414    learning rate
39        0.061     False    15.972236     0.740067  feature removal
36        0.099     False    17.179665     0.739731  feature removal
10        0.009     False    21.366778     0.737037          epsilon
45        0.066     False    17.222705     0.735690       feature *2
32        0.035     False    17.264986     0.729630  feature removal
4         6.472      True    46.838322     0.688552    learning rate
2         6.500      True   106.221172     0.683502    learning rate
1         8.592      True   118.189617     0.683165    learning rate
3         8.383      True    87.970034     0.683165    learning rate
9         0.000     False    20.585959     0.516835    learning rate
```

[29]:
```
#model_data.tail(1)
#model_data.drop(model_data.tail(1).index,inplace=True) # drop last n rows
show_sorted_model(model_data)
```

[29]:
```
   model_name                                     description  \
34         kd                 log age-insulin-no Heart Rate
40         kd                  log age-insulin-no heartrate
42         kd           log age-insulin-no heartrate-*2 Glucose
43         kd     log age-insulin-no heartrate-*2 BloodPressure
41         kd       log age-insulin-no heartrate-*2 Pregnancies
27         kd                               log age-insulin
30         kd                               log age-insulin
28         kd                           log age-insulin-Dpf
47         kd               log age-insulin-no heartrate-*2 BMI
46         kd           log age-insulin-no heartrate-*2 Insulin
23         kd                                   log Insulin
38         kd       log age-insulin-no DiabetesPedigreeFunction
25         kd                  log DiabetesPedigreeFunction
6          kd         whole model-lr:lr_type.iteration_plus_one
26         kd                                       log Age
14         kd                    whole model-epsilon:1e-06
16         kd                    whole model-epsilon:1e-08
15         kd                    whole model-epsilon:1e-07
17         kd                    whole model-epsilon:1e-09
33         kd                log age-insulin-no BloodPressure
```

```
31          kd                       log age-insulin-no Pregnancies
35          kd                     log age-insulin-no SkinThickness
5           kd                     whole model-lr:lr_type.iteration
44          kd           log age-insulin-no heartrate-*2 Heart Rate
18          kd                                      log Pregnancies
13          kd                            whole model-epsilon:1e-05
7           kd                   whole model-lr:lr_type.sample_size
0           kd                                          whole model
19          kd                                          log Glucose
12          kd                           whole model-epsilon:0.0001
24          kd                                              log BMI
37          kd                               log age-insulin-no BMI
22          kd                                    log SkinThickness
21          kd                                       log Heart Rate
29          kd                                         log skin-bmi
11          kd                            whole model-epsilon:0.001
48          kd  log age-insulin-no heartrate-*2 DiabetesPedigr…
20          kd                                    log BloodPressure
8           kd               whole model-lr:lr_type.ten_sample_size
39          kd                               log age-insulin-no Age
36          kd                           log age-insulin-no Insulin
10          kd                             whole model-epsilon:0.01
45          kd        log age-insulin-no heartrate-*2 SkinThickness
32          kd                           log age-insulin-no Glucose
4           kd                                whole model-lr:0.05
2           kd                                whole model-lr:0.25
1           kd                                 whole model-lr:0.5
3           kd                               whole model-lr:0.125
9           kd           whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
34   lr_type.iteration_plus_one       813
40   lr_type.iteration_plus_one       813
42   lr_type.iteration_plus_one       706
43   lr_type.iteration_plus_one       702
41   lr_type.iteration_plus_one       832
27   lr_type.iteration_plus_one       836
30   lr_type.iteration_plus_one       836
28   lr_type.iteration_plus_one       742
47   lr_type.iteration_plus_one       938
46   lr_type.iteration_plus_one       741
23   lr_type.iteration_plus_one       802
38   lr_type.iteration_plus_one       995
25   lr_type.iteration_plus_one      1118
6    lr_type.iteration_plus_one      1476
26   lr_type.iteration_plus_one      1425
14   lr_type.iteration_plus_one      1476
```

```
16   lr_type.iteration_plus_one      9021
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
33   lr_type.iteration_plus_one       656
31   lr_type.iteration_plus_one       877
35   lr_type.iteration_plus_one       987
5             lr_type.iteration      2220
44   lr_type.iteration_plus_one       834
18   lr_type.iteration_plus_one      1129
13   lr_type.iteration_plus_one       636
7           lr_type.sample_size      2048
0                          0.01       956
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
24   lr_type.iteration_plus_one      1341
37   lr_type.iteration_plus_one       711
22   lr_type.iteration_plus_one      1075
21   lr_type.iteration_plus_one      1008
29   lr_type.iteration_plus_one       978
11   lr_type.iteration_plus_one       165
48   lr_type.iteration_plus_one      1274
20   lr_type.iteration_plus_one      1045
8       lr_type.ten_sample_size      3516
39   lr_type.iteration_plus_one       790
36   lr_type.iteration_plus_one      1293
10   lr_type.iteration_plus_one        84
45   lr_type.iteration_plus_one      1507
32   lr_type.iteration_plus_one       484
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
3                         0.125    150000
9   lr_type.hundred_sample_size         2

                                              weights       epsilon  \
34  [[2.7129340471248065], [8.273684993858577], [-…  1.000000e-06
40  [[2.7129340471248065], [8.273684993858577], [-…  1.000000e-06
42  [[2.7707684939995394], [5.782931625664704], [-…  1.000000e-06
43  [[2.6840445568619615], [8.100362853592744], [-…  1.000000e-06
41  [[3.7667195356425127], [8.238912283417783], [-…  1.000000e-06
27  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
30  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
28  [[2.81722351586145], [8.11597903143088], [-0.8…  1.000000e-06
47  [[3.1911630539778977], [8.668229576381282], [-…  1.000000e-06
46  [[2.5822636064767592], [7.952642108792551], [-…  1.000000e-06
23  [[2.686500388124719], [8.005602930239544], [-0…  1.000000e-06
38  [[3.528294663986213], [8.888662016489207], [-0…  1.000000e-06
```

25	[[3.1317646256736045], [8.935591338567065], [-…	1.000000e-06
6	[[3.175241989157885], [9.517996847051375], [-1…	1.000000e-06
26	[[3.2317379647520714], [9.47235968785085], [-1…	1.000000e-06
14	[[3.175241989157885], [9.517996847051375], [-1…	1.000000e-06
16	[[2.96533714092659], [8.33217514426845], [-1.0…	1.000000e-08
15	[[3.0601381991207917], [8.83862515342757], [-1…	1.000000e-07
17	[[2.887385852612343], [7.9553323804450375], [-…	1.000000e-09
33	[[2.726699034298592], [7.741999408211606], [-0…	1.000000e-06
31	[[7.954655592251466], [-1.0448422866347953], […	1.000000e-06
35	[[2.8454163866127455], [7.444158178181417], [-…	1.000000e-06
5	[[3.6602142620593963], [11.56162794637247], [-…	1.000000e-06
44	[[2.724842549802258], [8.017986056469372], [-1…	1.000000e-06
18	[[0.47048312539781034], [8.606990965621858], […	1.000000e-06
13	[[3.3327577160627104], [10.39377495841507], [-…	1.000000e-05
7	[[2.5011580589018196], [6.514304926750582], [-…	1.000000e-06
0	[[2.5819277747484066], [6.780007402086465], [-…	1.000000e-06
19	[[2.7341191226158608], [1.1070301791999142], […	1.000000e-06
12	[[3.6109084400585387], [11.42702526345321], [-…	1.000000e-04
24	[[3.016669053499772], [9.333632805388874], [-1…	1.000000e-06
37	[[2.5109910019024677], [7.816325231383992], [-…	1.000000e-06
22	[[3.097120568287582], [8.175406022439612], [-1…	1.000000e-06
21	[[2.7030305464057154], [8.683157932519787], [-…	1.000000e-06
29	[[2.989559318463753], [8.044352212204572], [-1…	1.000000e-06
11	[[4.254111597053624], [12.705390512862264], [-…	1.000000e-03
48	[[2.8730033537390356], [8.894450845054584], [-…	1.000000e-06
20	[[3.0167390363517588], [8.607754261039704], [0…	1.000000e-06
8	[[1.8917011563673924], [3.7425174840678586], […	1.000000e-06
39	[[2.5616285768893263], [7.794299182419026], [-…	1.000000e-06
36	[[2.820836202474606], [9.936728220495777], [-0…	1.000000e-06
10	[[5.958742925531525], [14.327917968306403], [-…	1.000000e-02
45	[[2.8269443408264547], [10.095837895438326], […	1.000000e-06
32	[[2.360917183337187], [-0.8182987407890269], […	1.000000e-06
4	[[9.970728371948313], [26.491694165903258], [-…	1.000000e-06
2	[[47.64433276868854], [130.6790054571249], [-1…	1.000000e-06
1	[[95.81933692733199], [262.7825656994754], [-3…	1.000000e-06
3	[[24.070810464619214], [65.46666343519061], [-…	1.000000e-06
9	[[0.0003065942121212121], [0.00038259484848484…	1.000000e-06

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
34	0.062	False	15.717127	0.759933	feature removal
40	0.045	False	15.717127	0.759933	feature removal
42	0.032	False	15.705729	0.758586	feature *2
43	0.033	False	15.651793	0.758249	feature *2
41	0.046	False	15.915845	0.758249	feature *2
27	0.035	False	15.737679	0.757576	log
30	0.049	False	15.737679	0.757576	log
28	0.036	False	15.629731	0.756902	log

47	0.039	False	15.849282	0.756566	feature *2
46	0.031	False	15.840965	0.755556	feature *2
23	0.037	False	15.821280	0.755556	log
38	0.072	False	15.632706	0.751852	feature removal
25	0.051	False	16.145685	0.751852	log
6	0.105	False	16.436353	0.751515	learning rate
26	0.070	False	16.404716	0.751515	log
14	0.072	False	16.436353	0.751515	epsilon
16	0.413	False	16.097559	0.751515	epsilon
15	0.166	False	16.225842	0.751178	epsilon
17	0.940	False	16.019264	0.750842	epsilon
33	0.046	False	15.583091	0.750842	feature removal
31	0.041	False	16.310581	0.750842	feature removal
35	0.073	False	15.788618	0.750505	feature removal
5	0.162	False	17.234261	0.750505	learning rate
44	0.036	False	15.823767	0.750168	feature *2
18	0.048	False	16.526437	0.750168	log
13	0.027	False	16.775910	0.749495	epsilon
7	0.155	False	15.849570	0.749495	learning rate
0	0.049	False	15.855793	0.749495	all features
19	0.023	False	16.070200	0.748485	log
12	0.013	False	17.369442	0.747475	epsilon
24	0.056	False	16.234284	0.747138	log
37	0.049	False	15.701077	0.746465	feature removal
22	0.069	False	16.334522	0.746465	log
21	0.044	False	16.245876	0.744781	log
29	0.061	False	16.176293	0.744108	log
11	0.009	False	18.607920	0.743771	epsilon
48	0.055	False	16.098061	0.743771	feature *2
20	0.052	False	16.291746	0.743434	log
8	0.266	False	16.462365	0.741414	learning rate
39	0.061	False	15.972236	0.740067	feature removal
36	0.099	False	17.179665	0.739731	feature removal
10	0.009	False	21.366778	0.737037	epsilon
45	0.066	False	17.222705	0.735690	feature *2
32	0.035	False	17.264986	0.729630	feature removal
4	6.472	True	46.838322	0.688552	learning rate
2	6.500	True	106.221172	0.683502	learning rate
1	8.592	True	118.189617	0.683165	learning rate
3	8.383	True	87.970034	0.683165	learning rate
9	0.000	False	20.585959	0.516835	learning rate

[30]:
```
#check power 3 of features
for i in range(kd_x_train_features.shape[1]):
    print (i,',column=>',kd_columns[i])
    kd_x_power = power_n_feature(pd.DataFrame(kd_x_train_features),i,3).
 ↪to_numpy()
```

```
    title = 'log age-insulin-no heartrate-*3 {}'.format(kd_columns[i])
    print(title)
    model_data = run_model('kd',title,learning_rate = 0, learning_rate_type =␣
↪learning_rate_type.iteration_plus_one ,max_iterations = 150000,
                        epsilon = 1e-6,x_train = kd_x_power , y_train =␣
↪kd_y_train , model_data = model_data,variable = 'feature *3')
show_sorted_model(model_data)
```

```
0 ,column=> Pregnancies
log age-insulin-no heartrate-*3 Pregnancies
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
1 ,column=> Glucose
log age-insulin-no heartrate-*3 Glucose
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
2 ,column=> BloodPressure
log age-insulin-no heartrate-*3 BloodPressure
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
3 ,column=> Heart Rate
log age-insulin-no heartrate-*3 Heart Rate
i======> 0
```

```
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
4 ,column=> SkinThickness
log age-insulin-no heartrate-*3 SkinThickness
i======> 0
i======> 1
i======> 2
i======> 3
```

```
<ipython-input-8-caf8b4b6b9cd>:100: RuntimeWarning: overflow encountered in exp
  return 1 / (1 + np.exp(-arg))
```

```
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
5 ,column=> Insulin
log age-insulin-no heartrate-*3 Insulin
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
6 ,column=> BMI
log age-insulin-no heartrate-*3 BMI
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

```
7 ,column=> DiabetesPedigreeFunction
log age-insulin-no heartrate-*3 DiabetesPedigreeFunction
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[30]:

	model_name	description \
53	kd	log age-insulin-no heartrate-*3 SkinThickness
49	kd	log age-insulin-no heartrate-*3 Pregnancies
40	kd	log age-insulin-no heartrate
34	kd	log age-insulin-no Heart Rate
50	kd	log age-insulin-no heartrate-*3 Glucose
42	kd	log age-insulin-no heartrate-*2 Glucose
43	kd	log age-insulin-no heartrate-*2 BloodPressure
41	kd	log age-insulin-no heartrate-*2 Pregnancies
27	kd	log age-insulin
30	kd	log age-insulin
28	kd	log age-insulin-Dpf
47	kd	log age-insulin-no heartrate-*2 BMI
51	kd	log age-insulin-no heartrate-*3 BloodPressure
46	kd	log age-insulin-no heartrate-*2 Insulin
23	kd	log Insulin
55	kd	log age-insulin-no heartrate-*3 BMI
25	kd	log DiabetesPedigreeFunction
38	kd	log age-insulin-no DiabetesPedigreeFunction
16	kd	whole model-epsilon:1e-08
14	kd	whole model-epsilon:1e-06
6	kd	whole model-lr:lr_type.iteration_plus_one
26	kd	log Age
15	kd	whole model-epsilon:1e-07
17	kd	whole model-epsilon:1e-09
33	kd	log age-insulin-no BloodPressure
31	kd	log age-insulin-no Pregnancies
35	kd	log age-insulin-no SkinThickness
5	kd	whole model-lr:lr_type.iteration
44	kd	log age-insulin-no heartrate-*2 Heart Rate
18	kd	log Pregnancies
0	kd	whole model
13	kd	whole model-epsilon:1e-05
7	kd	whole model-lr:lr_type.sample_size

52	kd	log age-insulin-no heartrate-*3 Heart Rate
19	kd	log Glucose
12	kd	whole model-epsilon:0.0001
24	kd	log BMI
54	kd	log age-insulin-no heartrate-*3 Insulin
22	kd	log SkinThickness
37	kd	log age-insulin-no BMI
21	kd	log Heart Rate
29	kd	log skin-bmi
11	kd	whole model-epsilon:0.001
48	kd	log age-insulin-no heartrate-*2 DiabetesPedigr…
20	kd	log BloodPressure
8	kd	whole model-lr:lr_type.ten_sample_size
39	kd	log age-insulin-no Age
36	kd	log age-insulin-no Insulin
56	kd	log age-insulin-no heartrate-*3 DiabetesPedigr…
10	kd	whole model-epsilon:0.01
45	kd	log age-insulin-no heartrate-*2 SkinThickness
32	kd	log age-insulin-no Glucose
4	kd	whole model-lr:0.05
2	kd	whole model-lr:0.25
1	kd	whole model-lr:0.5
3	kd	whole model-lr:0.125
9	kd	whole model-lr:lr_type.hundred_sample_size

	learning_rate	iteration	\
53	lr_type.iteration_plus_one	877	
49	lr_type.iteration_plus_one	858	
40	lr_type.iteration_plus_one	813	
34	lr_type.iteration_plus_one	813	
50	lr_type.iteration_plus_one	708	
42	lr_type.iteration_plus_one	706	
43	lr_type.iteration_plus_one	702	
41	lr_type.iteration_plus_one	832	
27	lr_type.iteration_plus_one	836	
30	lr_type.iteration_plus_one	836	
28	lr_type.iteration_plus_one	742	
47	lr_type.iteration_plus_one	938	
51	lr_type.iteration_plus_one	695	
46	lr_type.iteration_plus_one	741	
23	lr_type.iteration_plus_one	802	
55	lr_type.iteration_plus_one	959	
25	lr_type.iteration_plus_one	1118	
38	lr_type.iteration_plus_one	995	
16	lr_type.iteration_plus_one	9021	
14	lr_type.iteration_plus_one	1476	
6	lr_type.iteration_plus_one	1476	

```
26   lr_type.iteration_plus_one      1425
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
33   lr_type.iteration_plus_one       656
31   lr_type.iteration_plus_one       877
35   lr_type.iteration_plus_one       987
5             lr_type.iteration      2220
44   lr_type.iteration_plus_one       834
18   lr_type.iteration_plus_one      1129
0                          0.01       956
13   lr_type.iteration_plus_one       636
7           lr_type.sample_size      2048
52   lr_type.iteration_plus_one       893
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
24   lr_type.iteration_plus_one      1341
54   lr_type.iteration_plus_one       750
22   lr_type.iteration_plus_one      1075
37   lr_type.iteration_plus_one       711
21   lr_type.iteration_plus_one      1008
29   lr_type.iteration_plus_one       978
11   lr_type.iteration_plus_one       165
48   lr_type.iteration_plus_one      1274
20   lr_type.iteration_plus_one      1045
8       lr_type.ten_sample_size      3516
39   lr_type.iteration_plus_one       790
36   lr_type.iteration_plus_one      1293
56   lr_type.iteration_plus_one      1714
10   lr_type.iteration_plus_one        84
45   lr_type.iteration_plus_one      1507
32   lr_type.iteration_plus_one       484
4                          0.05    150000
2                          0.25    150000
1                           0.5    150000
3                         0.125    150000
9   lr_type.hundred_sample_size         2

                                              weights       epsilon  \
53  [[2.7512737233256144], [8.386789991625008], [-…  1.000000e-06
49  [[3.733327173884576], [8.180972680577565], [-1…  1.000000e-06
40  [[2.7129340471248065], [8.273684993858577], [-…  1.000000e-06
34  [[2.7129340471248065], [8.273684993858577], [-…  1.000000e-06
50  [[2.8106015374663595], [5.369363601783941], [-…  1.000000e-06
42  [[2.7707684939995394], [5.782931625664704], [-…  1.000000e-06
43  [[2.6840445568619615], [8.100362853592744], [-…  1.000000e-06
41  [[3.7667195356425127], [8.238912283417783], [-…  1.000000e-06
27  [[2.7683606646024232], [8.122909449757739], [-…  1.000000e-06
```

30	[[2.7683606646024232], [8.122909449757739], [-…	1.000000e-06
28	[[2.81722351586145], [8.11597903143088], [-0.8…	1.000000e-06
47	[[3.1911630539778977], [8.668229576381282], [-…	1.000000e-06
51	[[2.6832157340323204], [8.090166152115357], [-…	1.000000e-06
46	[[2.5822636064767592], [7.952642108792551], [-…	1.000000e-06
23	[[2.686500388124719], [8.005602930239544], [-0…	1.000000e-06
55	[[3.3886056455906357], [8.833035181633305], [-…	1.000000e-06
25	[[3.1317646256736045], [8.935591338567065], [-…	1.000000e-06
38	[[3.528294663986213], [8.888662016489207], [-0…	1.000000e-06
16	[[2.96533714092659], [8.33217514426845], [-1.0…	1.000000e-08
14	[[3.175241989157885], [9.517996847051375], [-1…	1.000000e-06
6	[[3.175241989157885], [9.517996847051375], [-1…	1.000000e-06
26	[[3.2317379647520714], [9.47235968785085], [-1…	1.000000e-06
15	[[3.0601381991207917], [8.83862515342757], [-1…	1.000000e-07
17	[[2.887385852612343], [7.9553323804450375], [-…	1.000000e-09
33	[[2.726699034298592], [7.741999408211606], [-0…	1.000000e-06
31	[[7.954655592251466], [-1.0448422866347953], […	1.000000e-06
35	[[2.8454163866127455], [7.444158178181417], [-…	1.000000e-06
5	[[3.6602142620593963], [11.56162794637247], [-…	1.000000e-06
44	[[2.724842549802258], [8.017986056469372], [-1…	1.000000e-06
18	[[0.47048312539781034], [8.606990965621858], […	1.000000e-06
0	[[2.5819277747484066], [6.780007402086465], [-…	1.000000e-06
13	[[3.3327577160627104], [10.39377495841507], [-…	1.000000e-05
7	[[2.5011580589018196], [6.514304926750582], [-…	1.000000e-06
52	[[2.7265668621754484], [7.7797756044220066], […	1.000000e-06
19	[[2.7341191226158608], [1.1070301791999142], […	1.000000e-06
12	[[3.6109084400585387], [11.42702526345321], [-…	1.000000e-04
24	[[3.016669053499772], [9.333632805388874], [-1…	1.000000e-06
54	[[2.4931005393729446], [7.908737551062805], [-…	1.000000e-06
22	[[3.097120568287582], [8.175406022439612], [-1…	1.000000e-06
37	[[2.5109910019024677], [7.816325231383992], [-…	1.000000e-06
21	[[2.7030305464057154], [8.683157932519787], [-…	1.000000e-06
29	[[2.989559318463753], [8.044352212204572], [-1…	1.000000e-06
11	[[4.254111597053624], [12.705390512862264], [-…	1.000000e-03
48	[[2.8730033537390356], [8.894450845054584], [-…	1.000000e-06
20	[[3.0167390363517588], [8.607754261039704], [0…	1.000000e-06
8	[[1.8917011563673924], [3.7425174840678586], […	1.000000e-06
39	[[2.5616285768893263], [7.794299182419026], [-…	1.000000e-06
36	[[2.820836202474606], [9.936728220495777], [-0…	1.000000e-06
56	[[2.8470517371490716], [9.636035150531411], [-…	1.000000e-06
10	[[5.958742925531525], [14.327917968306403], [-…	1.000000e-02
45	[[2.8269443408264547], [10.095837895438326], […	1.000000e-06
32	[[2.360917183337187], [-0.8182987407890269], […	1.000000e-06
4	[[9.970728371948313], [26.491694165903258], [-…	1.000000e-06
2	[[47.64433276868854], [130.6790054571249], [-1…	1.000000e-06
1	[[95.81933692733199], [262.7825656994754], [-3…	1.000000e-06
3	[[24.070810464619214], [65.46666343519061], [-…	1.000000e-06

9 [[0.0003065942121212121], [0.00038259484848484… 1.000000e-06

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
53	0.058	False	15.777801	0.773737	feature *3
49	0.043	False	16.092013	0.761279	feature *3
40	0.045	False	15.717127	0.759933	feature removal
34	0.062	False	15.717127	0.759933	feature removal
50	0.035	False	15.784424	0.758923	feature *3
42	0.032	False	15.705729	0.758586	feature *2
43	0.033	False	15.651793	0.758249	feature *2
41	0.046	False	15.915845	0.758249	feature *2
27	0.035	False	15.737679	0.757576	log
30	0.049	False	15.737679	0.757576	log
28	0.036	False	15.629731	0.756902	log
47	0.039	False	15.849282	0.756566	feature *2
51	0.039	False	15.634238	0.756229	feature *3
46	0.031	False	15.840965	0.755556	feature *2
23	0.037	False	15.821280	0.755556	log
55	0.045	False	15.814899	0.755219	feature *3
25	0.051	False	16.145685	0.751852	log
38	0.072	False	15.632706	0.751852	feature removal
16	0.413	False	16.097559	0.751515	epsilon
14	0.072	False	16.436353	0.751515	epsilon
6	0.105	False	16.436353	0.751515	learning rate
26	0.070	False	16.404716	0.751515	log
15	0.166	False	16.225842	0.751178	epsilon
17	0.940	False	16.019264	0.750842	epsilon
33	0.046	False	15.583091	0.750842	feature removal
31	0.041	False	16.310581	0.750842	feature removal
35	0.073	False	15.788618	0.750505	feature removal
5	0.162	False	17.234261	0.750505	learning rate
44	0.036	False	15.823767	0.750168	feature *2
18	0.048	False	16.526437	0.750168	log
0	0.049	False	15.855793	0.749495	all features
13	0.027	False	16.775910	0.749495	epsilon
7	0.155	False	15.849570	0.749495	learning rate
52	0.044	False	15.929618	0.748822	feature *3
19	0.023	False	16.070200	0.748485	log
12	0.013	False	17.369442	0.747475	epsilon
24	0.056	False	16.234284	0.747138	log
54	0.036	False	15.882348	0.746801	feature *3
22	0.069	False	16.334522	0.746465	log
37	0.049	False	15.701077	0.746465	feature removal
21	0.044	False	16.245876	0.744781	log
29	0.061	False	16.176293	0.744108	log
11	0.009	False	18.607920	0.743771	epsilon
48	0.055	False	16.098061	0.743771	feature *2

```
20          0.052          False    16.291746        0.743434              log
8           0.266          False    16.462365        0.741414    learning rate
39          0.061          False    15.972236        0.740067  feature removal
36          0.099          False    17.179665        0.739731  feature removal
56          0.081          False    16.352570        0.739394       feature *3
10          0.009          False    21.366778        0.737037          epsilon
45          0.066          False    17.222705        0.735690       feature *2
32          0.035          False    17.264986        0.729630  feature removal
4           6.472           True    46.838322        0.688552    learning rate
2           6.500           True   106.221172        0.683502    learning rate
1           8.592           True   118.189617        0.683165    learning rate
3           8.383           True    87.970034        0.683165    learning rate
9           0.000          False    20.585959        0.516835    learning rate
```

[31]:
```
(model_data.sort_values(by=['accuracy_kfold'], ascending=False)).
↪to_csv('kidney_disease_models.csv', index=False)
```

[32]:
```
import os
cwd = os.getcwd()
print(cwd)
```

```
/content
```

[33]:
```
def recursive_feature_elimination(X, y, model, num_features):
    num_samples, num_total_features = X.shape

    # Initialize the mask to include all features
    mask = np.ones(num_total_features, dtype=bool)

    def rfe(X, y, model, mask, num_features):

        if np.sum(mask) == num_features:
            return X[:, mask]

        model.fit(X[:, mask], y)
        feature_importances = np.zeros(num_total_features)
        feature_importances[mask] = model.feature_importances_

        least_important_feature_idx = np.argmin(feature_importances)
        mask[least_important_feature_idx] = False
        return rfe(X, y, model, mask, num_features)
    return rfe(X, y, model, mask, num_features)
```

[34]:
```
(model_data.sort_values(by=['accuracy_kfold'], ascending=False))
```

[34]:
```
   model_name                                       description  \
53         kd      log age-insulin-no heartrate-*3 SkinThickness
```

49	kd	log age-insulin-no heartrate-*3 Pregnancies
40	kd	log age-insulin-no heartrate
34	kd	log age-insulin-no Heart Rate
50	kd	log age-insulin-no heartrate-*3 Glucose
42	kd	log age-insulin-no heartrate-*2 Glucose
43	kd	log age-insulin-no heartrate-*2 BloodPressure
41	kd	log age-insulin-no heartrate-*2 Pregnancies
27	kd	log age-insulin
30	kd	log age-insulin
28	kd	log age-insulin-Dpf
47	kd	log age-insulin-no heartrate-*2 BMI
51	kd	log age-insulin-no heartrate-*3 BloodPressure
46	kd	log age-insulin-no heartrate-*2 Insulin
23	kd	log Insulin
55	kd	log age-insulin-no heartrate-*3 BMI
25	kd	log DiabetesPedigreeFunction
38	kd	log age-insulin-no DiabetesPedigreeFunction
16	kd	whole model-epsilon:1e-08
14	kd	whole model-epsilon:1e-06
6	kd	whole model-lr:lr_type.iteration_plus_one
26	kd	log Age
15	kd	whole model-epsilon:1e-07
17	kd	whole model-epsilon:1e-09
33	kd	log age-insulin-no BloodPressure
31	kd	log age-insulin-no Pregnancies
35	kd	log age-insulin-no SkinThickness
5	kd	whole model-lr:lr_type.iteration
44	kd	log age-insulin-no heartrate-*2 Heart Rate
18	kd	log Pregnancies
0	kd	whole model
13	kd	whole model-epsilon:1e-05
7	kd	whole model-lr:lr_type.sample_size
52	kd	log age-insulin-no heartrate-*3 Heart Rate
19	kd	log Glucose
12	kd	whole model-epsilon:0.0001
24	kd	log BMI
54	kd	log age-insulin-no heartrate-*3 Insulin
22	kd	log SkinThickness
37	kd	log age-insulin-no BMI
21	kd	log Heart Rate
29	kd	log skin-bmi
11	kd	whole model-epsilon:0.001
48	kd	log age-insulin-no heartrate-*2 DiabetesPedigr…
20	kd	log BloodPressure
8	kd	whole model-lr:lr_type.ten_sample_size
39	kd	log age-insulin-no Age
36	kd	log age-insulin-no Insulin

```
56         kd  log age-insulin-no heartrate-*3 DiabetesPedigr…
10         kd                           whole model-epsilon:0.01
45         kd      log age-insulin-no heartrate-*2 SkinThickness
32         kd                         log age-insulin-no Glucose
4          kd                                whole model-lr:0.05
2          kd                                whole model-lr:0.25
1          kd                                 whole model-lr:0.5
3          kd                               whole model-lr:0.125
9          kd         whole model-lr:lr_type.hundred_sample_size

                learning_rate iteration  \
53   lr_type.iteration_plus_one       877
49   lr_type.iteration_plus_one       858
40   lr_type.iteration_plus_one       813
34   lr_type.iteration_plus_one       813
50   lr_type.iteration_plus_one       708
42   lr_type.iteration_plus_one       706
43   lr_type.iteration_plus_one       702
41   lr_type.iteration_plus_one       832
27   lr_type.iteration_plus_one       836
30   lr_type.iteration_plus_one       836
28   lr_type.iteration_plus_one       742
47   lr_type.iteration_plus_one       938
51   lr_type.iteration_plus_one       695
46   lr_type.iteration_plus_one       741
23   lr_type.iteration_plus_one       802
55   lr_type.iteration_plus_one       959
25   lr_type.iteration_plus_one      1118
38   lr_type.iteration_plus_one       995
16   lr_type.iteration_plus_one      9021
14   lr_type.iteration_plus_one      1476
6    lr_type.iteration_plus_one      1476
26   lr_type.iteration_plus_one      1425
15   lr_type.iteration_plus_one      3643
17   lr_type.iteration_plus_one     22164
33   lr_type.iteration_plus_one       656
31   lr_type.iteration_plus_one       877
35   lr_type.iteration_plus_one       987
5             lr_type.iteration      2220
44   lr_type.iteration_plus_one       834
18   lr_type.iteration_plus_one      1129
0                          0.01       956
13   lr_type.iteration_plus_one       636
7           lr_type.sample_size      2048
52   lr_type.iteration_plus_one       893
19   lr_type.iteration_plus_one       560
12   lr_type.iteration_plus_one       314
```

24	lr_type.iteration_plus_one	1341
54	lr_type.iteration_plus_one	750
22	lr_type.iteration_plus_one	1075
37	lr_type.iteration_plus_one	711
21	lr_type.iteration_plus_one	1008
29	lr_type.iteration_plus_one	978
11	lr_type.iteration_plus_one	165
48	lr_type.iteration_plus_one	1274
20	lr_type.iteration_plus_one	1045
8	lr_type.ten_sample_size	3516
39	lr_type.iteration_plus_one	790
36	lr_type.iteration_plus_one	1293
56	lr_type.iteration_plus_one	1714
10	lr_type.iteration_plus_one	84
45	lr_type.iteration_plus_one	1507
32	lr_type.iteration_plus_one	484
4	0.05	150000
2	0.25	150000
1	0.5	150000
3	0.125	150000
9	lr_type.hundred_sample_size	2

	weights	epsilon \
53	[[2.7512737233256144], [8.386789991625008], [-…	1.000000e-06
49	[[3.733327173884576], [8.180972680577565], [-1…	1.000000e-06
40	[[2.7129340471248065], [8.273684993858577], [-…	1.000000e-06
34	[[2.7129340471248065], [8.273684993858577], [-…	1.000000e-06
50	[[2.8106015374663595], [5.369363601783941], [-…	1.000000e-06
42	[[2.7707684939995394], [5.782931625664704], [-…	1.000000e-06
43	[[2.6840445568619615], [8.100362853592744], [-…	1.000000e-06
41	[[3.7667195356425127], [8.238912283417783], [-…	1.000000e-06
27	[[2.7683606646024232], [8.122909449757739], [-…	1.000000e-06
30	[[2.7683606646024232], [8.122909449757739], [-…	1.000000e-06
28	[[2.81722351586145], [8.11597903143088], [-0.8…	1.000000e-06
47	[[3.1911630539778977], [8.668229576381282], [-…	1.000000e-06
51	[[2.6832157340323204], [8.090166152115357], [-…	1.000000e-06
46	[[2.5822636064767592], [7.952642108792551], [-…	1.000000e-06
23	[[2.686500388124719], [8.005602930239544], [-0…	1.000000e-06
55	[[3.3886056455906357], [8.833035181633305], [-…	1.000000e-06
25	[[3.1317646256736045], [8.935591338567065], [-…	1.000000e-06
38	[[3.528294663986213], [8.888662016489207], [-0…	1.000000e-06
16	[[2.96533714092659], [8.33217514426845], [-1.0…	1.000000e-08
14	[[3.175241989157885], [9.517996847051375], [-1…	1.000000e-06
6	[[3.175241989157885], [9.517996847051375], [-1…	1.000000e-06
26	[[3.2317379647520714], [9.47235968785085], [-1…	1.000000e-06
15	[[3.0601381991207917], [8.83862515342757], [-1…	1.000000e-07
17	[[2.887385852612343], [7.9553323804450375], [-…	1.000000e-09

33	[[2.726699034298592], [7.741999408211606], [-0…	1.000000e-06
31	[[7.954655592251466], [-1.0448422866347953], […	1.000000e-06
35	[[2.8454163866127455], [7.444158178181417], [-…	1.000000e-06
5	[[3.6602142620593963], [11.56162794637247], [-…	1.000000e-06
44	[[2.724842549802258], [8.017986056469372], [-1…	1.000000e-06
18	[[0.47048312539781034], [8.606990965621858], […	1.000000e-06
0	[[2.5819277747484066], [6.780007402086465], [-…	1.000000e-06
13	[[3.3327577160627104], [10.39377495841507], [-…	1.000000e-05
7	[[2.5011580589018196], [6.514304926750582], [-…	1.000000e-06
52	[[2.7265668621754484], [7.7797756044220066], […	1.000000e-06
19	[[2.7341191226158608], [1.1070301791999142], […	1.000000e-06
12	[[3.6109084400585387], [11.42702526345321], [-…	1.000000e-04
24	[[3.016669053499772], [9.333632805388874], [-1…	1.000000e-06
54	[[2.4931005393729446], [7.908737551062805], [-…	1.000000e-06
22	[[3.097120568287582], [8.175406022439612], [-1…	1.000000e-06
37	[[2.5109910019024677], [7.816325231383992], [-…	1.000000e-06
21	[[2.7030305464057154], [8.683157932519787], [-…	1.000000e-06
29	[[2.989559318463753], [8.044352212204572], [-1…	1.000000e-06
11	[[4.254111597053624], [12.705390512862264], [-…	1.000000e-03
48	[[2.8730033537390356], [8.894450845054584], [-…	1.000000e-06
20	[[3.0167390363517588], [8.607754261039704], [0…	1.000000e-06
8	[[1.8917011563673924], [3.7425174840678586], […	1.000000e-06
39	[[2.5616285768893263], [7.794299182419026], [-…	1.000000e-06
36	[[2.820836202474606], [9.936728220495777], [-0…	1.000000e-06
56	[[2.8470517371490716], [9.636035150531411], [-…	1.000000e-06
10	[[5.958742925531525], [14.327917968306403], [-…	1.000000e-02
45	[[2.8269443408264547], [10.095837895438326], […	1.000000e-06
32	[[2.360917183337187], [-0.8182987407890269], […	1.000000e-06
4	[[9.970728371948313], [26.491694165903258], [-…	1.000000e-06
2	[[47.64433276868854], [130.6790054571249], [-1…	1.000000e-06
1	[[95.81933692733199], [262.7825656994754], [-3…	1.000000e-06
3	[[24.070810464619214], [65.46666343519061], [-…	1.000000e-06
9	[[0.0003065942121212121], [0.00038259484848484…	1.000000e-06

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
53	0.058	False	15.777801	0.773737	feature *3
49	0.043	False	16.092013	0.761279	feature *3
40	0.045	False	15.717127	0.759933	feature removal
34	0.062	False	15.717127	0.759933	feature removal
50	0.035	False	15.784424	0.758923	feature *3
42	0.032	False	15.705729	0.758586	feature *2
43	0.033	False	15.651793	0.758249	feature *2
41	0.046	False	15.915845	0.758249	feature *2
27	0.035	False	15.737679	0.757576	log
30	0.049	False	15.737679	0.757576	log
28	0.036	False	15.629731	0.756902	log
47	0.039	False	15.849282	0.756566	feature *2

51	0.039	False	15.634238	0.756229	feature *3
46	0.031	False	15.840965	0.755556	feature *2
23	0.037	False	15.821280	0.755556	log
55	0.045	False	15.814899	0.755219	feature *3
25	0.051	False	16.145685	0.751852	log
38	0.072	False	15.632706	0.751852	feature removal
16	0.413	False	16.097559	0.751515	epsilon
14	0.072	False	16.436353	0.751515	epsilon
6	0.105	False	16.436353	0.751515	learning rate
26	0.070	False	16.404716	0.751515	log
15	0.166	False	16.225842	0.751178	epsilon
17	0.940	False	16.019264	0.750842	epsilon
33	0.046	False	15.583091	0.750842	feature removal
31	0.041	False	16.310581	0.750842	feature removal
35	0.073	False	15.788618	0.750505	feature removal
5	0.162	False	17.234261	0.750505	learning rate
44	0.036	False	15.823767	0.750168	feature *2
18	0.048	False	16.526437	0.750168	log
0	0.049	False	15.855793	0.749495	all features
13	0.027	False	16.775910	0.749495	epsilon
7	0.155	False	15.849570	0.749495	learning rate
52	0.044	False	15.929618	0.748822	feature *3
19	0.023	False	16.070200	0.748485	log
12	0.013	False	17.369442	0.747475	epsilon
24	0.056	False	16.234284	0.747138	log
54	0.036	False	15.882348	0.746801	feature *3
22	0.069	False	16.334522	0.746465	log
37	0.049	False	15.701077	0.746465	feature removal
21	0.044	False	16.245876	0.744781	log
29	0.061	False	16.176293	0.744108	log
11	0.009	False	18.607920	0.743771	epsilon
48	0.055	False	16.098061	0.743771	feature *2
20	0.052	False	16.291746	0.743434	log
8	0.266	False	16.462365	0.741414	learning rate
39	0.061	False	15.972236	0.740067	feature removal
36	0.099	False	17.179665	0.739731	feature removal
56	0.081	False	16.352570	0.739394	feature *3
10	0.009	False	21.366778	0.737037	epsilon
45	0.066	False	17.222705	0.735690	feature *2
32	0.035	False	17.264986	0.729630	feature removal
4	6.472	True	46.838322	0.688552	learning rate
2	6.500	True	106.221172	0.683502	learning rate
1	8.592	True	118.189617	0.683165	learning rate
3	8.383	True	87.970034	0.683165	learning rate
9	0.000	False	20.585959	0.516835	learning rate

[47]:

```
#(model_data['variable']=='learning rate')
model_data.loc[model_data['variable'] == 'feature removal']
#model_data.loc[model_data['variable'] == 'all features']

#.sort_values(by=['accuracy_kfold'], ascending=False))
```

[47]:

```
   model_name                                  description  \
31         kd                log age-insulin-no Pregnancies
32         kd                    log age-insulin-no Glucose
33         kd              log age-insulin-no BloodPressure
34         kd                 log age-insulin-no Heart Rate
35         kd              log age-insulin-no SkinThickness
36         kd                    log age-insulin-no Insulin
37         kd                        log age-insulin-no BMI
38         kd  log age-insulin-no DiabetesPedigreeFunction
39         kd                        log age-insulin-no Age
40         kd                  log age-insulin-no heartrate

                 learning_rate iteration  \
31  lr_type.iteration_plus_one       877
32  lr_type.iteration_plus_one       484
33  lr_type.iteration_plus_one       656
34  lr_type.iteration_plus_one       813
35  lr_type.iteration_plus_one       987
36  lr_type.iteration_plus_one      1293
37  lr_type.iteration_plus_one       711
38  lr_type.iteration_plus_one       995
39  lr_type.iteration_plus_one       790
40  lr_type.iteration_plus_one       813

                                              weights   epsilon  elapsed_time  \
31  [[7.954655592251466], [-1.0448422866347953], […  0.000001         0.041
32  [[2.360917183337187], [-0.8182987407890269], […  0.000001         0.035
33  [[2.726699034298592], [7.741999408211606], [-0…  0.000001         0.046
34  [[2.7129340471248065], [8.273684993858577], [-…  0.000001         0.062
35  [[2.8454163866127455], [7.444158178181417], [-…  0.000001         0.073
36  [[2.820836202474606], [9.936728220495777], [-0…  0.000001         0.099
37  [[2.5109910019024677], [7.816325231383992], [-…  0.000001         0.049
38  [[3.528294663986213], [8.888662016489207], [-0…  0.000001         0.072
39  [[2.5616285768893263], [7.794299182419026], [-…  0.000001         0.061
40  [[2.7129340471248065], [8.273684993858577], [-…  0.000001         0.045

   is_max_reached       loss  accuracy_kfold         variable
31          False  16.310581        0.750842  feature removal
32          False  17.264986        0.729630  feature removal
33          False  15.583091        0.750842  feature removal
34          False  15.717127        0.759933  feature removal
```

35	False	15.788618	0.750505	feature removal
36	False	17.179665	0.739731	feature removal
37	False	15.701077	0.746465	feature removal
38	False	15.632706	0.751852	feature removal
39	False	15.972236	0.740067	feature removal
40	False	15.717127	0.759933	feature removal

whitewhine_model

February 19, 2023

[66]:
```
#delete_model(model_data)
show_sorted_model(model_data)
#print(kd_x_train.shape)
#delete_model(model_data)
```

[66]:
```
Empty DataFrame
Columns: [model_name, description, learning_rate, iteration, weights, epsilon,
elapsed_time, is_max_reached, loss, accuracy_kfold, variable]
Index: []
```

[67]:
```
model_data = run_model('ww','whole model',learning_rate = 0, learning_rate_type␣
↪= learning_rate_type.iteration ,max_iterations = 150000,
                    epsilon = 1e-5,x_train = ww_x_train , y_train =␣
↪ww_y_train , model_data = model_data,variable = 'all features')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[67]:
```
  model_name  description      learning_rate iteration  \
0         ww  whole model  lr_type.iteration      4032

                                             weights  epsilon  elapsed_time  \
0  [[16.378906342486571375], [-5.5259574517590583…  0.00001         5.061

  is_max_reached       loss  accuracy_kfold      variable
0          False  97.099984        0.735186  all features
```

[68]:
```
#explore different constant learning rates
#learning_rates = [1/2,1/4,0.01]
#for i in range(len(learning_rates)):
 # title = 'whole model-lr:' + str(learning_rates[i])
 # model_data = run_model('ww',title,learning_rate = learning_rates[i],␣
 ↪learning_rate_type = learning_rate_type.independent ,max_iterations = 150000,
  #                     epsilon = 1e-5,x_train = ww_x_train , y_train =␣
 ↪ww_y_train , model_data = model_data,variable = 'learning rate')
#show_sorted_model(model_data)
#very time consuming
```

[69]:
```
#explore different dependent learning rates
learning_rates_types = [learning_rate_type.iteration, learning_rate_type.
 ↪iteration_plus_one,learning_rate_type.sample_size,
                        learning_rate_type.ten_sample_size,learning_rate_type.
 ↪hundred_sample_size]
for i in range(len(learning_rates_types)):
  title = 'whole model-lr:' + str(learning_rates_types[i])
  model_data = run_model('ww',title,learning_rate = 0, learning_rate_type =␣
 ↪learning_rates_types[i] ,max_iterations = 150000,
                       epsilon = 1e-5,x_train = ww_x_train , y_train =␣
 ↪ww_y_train , model_data = model_data,variable = 'learning rate')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
```

```
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[69]:

```
  model_name                                    description  \
2         ww   whole model-lr:lr_type.iteration_plus_one
0         ww                                    whole model
1         ww              whole model-lr:lr_type.iteration
3         ww            whole model-lr:lr_type.sample_size
4         ww        whole model-lr:lr_type.ten_sample_size
5         ww  whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
2   lr_type.iteration_plus_one      2753
0            lr_type.iteration      4032
1            lr_type.iteration      4032
3          lr_type.sample_size      1362
4      lr_type.ten_sample_size        74
5  lr_type.hundred_sample_size         2

                                             weights  epsilon  elapsed_time  \
2  [[12.137409248966954996 4], [-4.520518798900213…  0.00001         1.110
0  [[16.378906342486571375], [-5.5259574517590583…  0.00001         5.061
```

```
1  [[16.378906342486571375], [-5.5259574517590583…  0.00001         1.660
3  [[3.239870586975073716], [-3.93183866315446415…  0.00001         0.605
4  [[0.07851626733184168337], [-0.114624863637962…  0.00001         0.044
5  [[0.00018731212820512820378], [-9.962399624765…  0.00001         0.001

  is_max_reached       loss  accuracy_kfold       variable
2          False  86.235038        0.738105  learning rate
0          False  97.099984        0.735186   all features
1          False  97.099984        0.735186  learning rate
3          False  80.126535        0.729210  learning rate
4          False  98.565776        0.536584  learning rate
5          False  99.809458        0.535056  learning rate
```

[70]:
```python
#explore different epsilons
epsilon_list = [1e-2,1e-3,1e-4,1e-5,1e-6,1e-7,1e-8,1e-9]
for i in range(len(epsilon_list)):
  title = 'whole model-epsilon:' + str(epsilon_list[i])
  model_data = run_model('ww',title,learning_rate = 0, learning_rate_type =␣
 ↪learning_rate_type.iteration_plus_one ,max_iterations = 150000,
                         epsilon = epsilon_list[i],x_train = ww_x_train , y_train␣
 ↪= ww_y_train , model_data = model_data,variable = 'epsilon')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
```

```
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
i======> 0
i======> 1
```

```
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[70]:

```
   model_name                                    description  \
11         ww                     whole model-epsilon:1e-07
12         ww                     whole model-epsilon:1e-08
2          ww   whole model-lr:lr_type.iteration_plus_one
9          ww                     whole model-epsilon:1e-05
13         ww                     whole model-epsilon:1e-09
10         ww                     whole model-epsilon:1e-06
0          ww                                   whole model
1          ww              whole model-lr:lr_type.iteration
8          ww                    whole model-epsilon:0.0001
6          ww                      whole model-epsilon:0.01
3          ww            whole model-lr:lr_type.sample_size
7          ww                     whole model-epsilon:0.001
4          ww        whole model-lr:lr_type.ten_sample_size
5          ww  whole model-lr:lr_type.hundred_sample_size

                    learning_rate  iteration  \
11    lr_type.iteration_plus_one      10464
12    lr_type.iteration_plus_one      19610
2     lr_type.iteration_plus_one       2753
9     lr_type.iteration_plus_one       2753
13    lr_type.iteration_plus_one      36990
10    lr_type.iteration_plus_one       5496
0              lr_type.iteration       4032
1              lr_type.iteration       4032
8     lr_type.iteration_plus_one       1263
6     lr_type.iteration_plus_one        235
3            lr_type.sample_size       1362
7     lr_type.iteration_plus_one        531
4        lr_type.ten_sample_size         74
5    lr_type.hundred_sample_size          2

                                              weights       epsilon  \
11  [[9.087798687760632232], [-4.04964890730847365…  1.000000e-07
12  [[8.305635754643900189], [-4.08145059850403449…  1.000000e-08
2   [[12.137409248966954996 4], [-4.520518798900213…  1.000000e-05
9   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
13  [[7.781465198825748270 8], [-4.1149838488482304…  1.000000e-09
```

```
10  [[10.283574906574218732], [-4.1060086401810997…  1.000000e-06
0   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
1   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
8   [[14.944116734687297383], [-5.9888301416859652…  1.000000e-04
6   [[20.268681216307229112], [-17.891141982398493…  1.000000e-02
3   [[3.239870586975073716], [-3.93183866315446415…  1.000000e-05
7   [[18.400300481674902137], [-9.9749477354584466…  1.000000e-03
4   [[0.07851626733184168337], [-0.114624863637962…  1.000000e-05
5   [[0.00018731212820512820378], [-9.962399624765…  1.000000e-05

    elapsed_time is_max_reached        loss  accuracy_kfold       variable
11         4.588          False   80.926339        0.739631        epsilon
12         7.677          False   80.216146        0.739354        epsilon
2          1.110          False   86.235038        0.738105  learning rate
9          1.104          False   86.235038        0.738105        epsilon
13        15.933          False   79.885937        0.738036        epsilon
10         2.197          False   82.538558        0.737964        epsilon
0          5.061          False   97.099984        0.735186   all features
1          1.660          False   97.099984        0.735186  learning rate
8          0.522          False   94.274794        0.734561        epsilon
6          0.151          False  134.658479        0.729835        epsilon
3          0.605          False   80.126535        0.729210  learning rate
7          0.365          False  109.385158        0.728721        epsilon
4          0.044          False   98.565776        0.536584  learning rate
5          0.001          False   99.809458        0.535056  learning rate
```

[71]:
```
ww_x_train_np = pd.DataFrame(ww_x_train)
```

[72]:
```
show_sorted_model(model_data)
```

[72]:
```
   model_name                                 description  \
11         ww                   whole model-epsilon:1e-07
12         ww                   whole model-epsilon:1e-08
2          ww   whole model-lr:lr_type.iteration_plus_one
9          ww                   whole model-epsilon:1e-05
13         ww                   whole model-epsilon:1e-09
10         ww                   whole model-epsilon:1e-06
0          ww                                 whole model
1          ww              whole model-lr:lr_type.iteration
8          ww                  whole model-epsilon:0.0001
6          ww                    whole model-epsilon:0.01
3          ww            whole model-lr:lr_type.sample_size
7          ww                   whole model-epsilon:0.001
4          ww        whole model-lr:lr_type.ten_sample_size
5          ww    whole model-lr:lr_type.hundred_sample_size

                   learning_rate iteration  \
```

11	lr_type.iteration_plus_one	10464
12	lr_type.iteration_plus_one	19610
2	lr_type.iteration_plus_one	2753
9	lr_type.iteration_plus_one	2753
13	lr_type.iteration_plus_one	36990
10	lr_type.iteration_plus_one	5496
0	lr_type.iteration	4032
1	lr_type.iteration	4032
8	lr_type.iteration_plus_one	1263
6	lr_type.iteration_plus_one	235
3	lr_type.sample_size	1362
7	lr_type.iteration_plus_one	531
4	lr_type.ten_sample_size	74
5	lr_type.hundred_sample_size	2

	weights	epsilon \
11	[[9.087798687760632232], [-4.04964890730847365…	1.000000e-07
12	[[8.305635754643900189], [-4.08145059850403449…	1.000000e-08
2	[[12.137409248966954996 4], [-4.520518798900213…	1.000000e-05
9	[[12.1374092489669549964], [-4.520518798900213…	1.000000e-05
13	[[7.781465198825748270 8], [-4.1149838488482304…	1.000000e-09
10	[[10.283574906574218732], [-4.1060086401810997…	1.000000e-06
0	[[16.378906342486571375], [-5.5259574517590583…	1.000000e-05
1	[[16.378906342486571375], [-5.5259574517590583…	1.000000e-05
8	[[14.944116734687297383], [-5.9888301416859652…	1.000000e-04
6	[[20.268681216307229112], [-17.891141982398493…	1.000000e-02
3	[[3.239870586975073716], [-3.93183866315446415…	1.000000e-05
7	[[18.400300481674902137], [-9.9749477354584466…	1.000000e-03
4	[[0.078516267331841683 37], [-0.114624863637962…	1.000000e-05
5	[[0.00018731212820512820378], [-9.962399624765…	1.000000e-05

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
11	4.588	False	80.926339	0.739631	epsilon
12	7.677	False	80.216146	0.739354	epsilon
2	1.110	False	86.235038	0.738105	learning rate
9	1.104	False	86.235038	0.738105	epsilon
13	15.933	False	79.885937	0.738036	epsilon
10	2.197	False	82.538558	0.737964	epsilon
0	5.061	False	97.099984	0.735186	all features
1	1.660	False	97.099984	0.735186	learning rate
8	0.522	False	94.274794	0.734561	epsilon
6	0.151	False	134.658479	0.729835	epsilon
3	0.605	False	80.126535	0.729210	learning rate
7	0.365	False	109.385158	0.728721	epsilon
4	0.044	False	98.565776	0.536584	learning rate
5	0.001	False	99.809458	0.535056	learning rate

[73]:

```
#explore different logs of features
for i in range(ww_x_train.shape[1]):
    print (i,',log , column=>',ww_columns[i])
    ww_x_train_modified = log_transform_normalize(ww_x_train_np,i)
    title = 'log {}'.format(ww_columns[i])
    model_data = run_model('ww',title,learning_rate = 0, learning_rate_type =␣
 ↪learning_rate_type.iteration_plus_one ,max_iterations = 150000,
                          epsilon = 1e-5,x_train = ww_x_train_modified , y_train =␣
 ↪ww_y_train , model_data = model_data,variable = 'log')
show_sorted_model(model_data)
```

```
0 ,log , column=> Alcohol
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
1 ,log , column=> Malic acid
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
2 ,log , column=> Ash
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
3 ,log , column=> Alkalinity of ash
i======> 0
i======> 1
```

```
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
4 ,log , column=> Magnesium
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
5 ,log , column=> Total phenols
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
6 ,log , column=> Flavanoids
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
7 ,log , column=> Nonflavanoid phenols
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
```

```
i======> 6
i======> 7
i======> 8
i======> 9
8 ,log , column=> Proanthocyanins
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
9 ,log , column=> Hue
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[73]:

```
   model_name                                description  \
19         ww                          log Total phenols
11         ww                 whole model-epsilon:1e-07
12         ww                 whole model-epsilon:1e-08
22         ww                        log Proanthocyanins
9          ww                 whole model-epsilon:1e-05
2          ww   whole model-lr:lr_type.iteration_plus_one
13         ww                 whole model-epsilon:1e-09
10         ww                 whole model-epsilon:1e-06
23         ww                                    log Hue
18         ww                              log Magnesium
21         ww                   log Nonflavanoid phenols
1          ww            whole model-lr:lr_type.iteration
0          ww                                whole model
15         ww                             log Malic acid
8          ww                whole model-epsilon:0.0001
16         ww                                    log Ash
20         ww                             log Flavanoids
17         ww                      log Alkalinity of ash
6          ww                  whole model-epsilon:0.01
```

```
3          ww          whole model-lr:lr_type.sample_size
7          ww                   whole model-epsilon:0.001
14         ww                                 log Alcohol
4          ww      whole model-lr:lr_type.ten_sample_size
5          ww  whole model-lr:lr_type.hundred_sample_size

                  learning_rate iteration  \
19   lr_type.iteration_plus_one      2743
11   lr_type.iteration_plus_one     10464
12   lr_type.iteration_plus_one     19610
22   lr_type.iteration_plus_one      2439
9    lr_type.iteration_plus_one      2753
2    lr_type.iteration_plus_one      2753
13   lr_type.iteration_plus_one     36990
10   lr_type.iteration_plus_one      5496
23   lr_type.iteration_plus_one      2118
18   lr_type.iteration_plus_one      2205
21   lr_type.iteration_plus_one      1844
1             lr_type.iteration      4032
0             lr_type.iteration      4032
15   lr_type.iteration_plus_one      2115
8    lr_type.iteration_plus_one      1263
16   lr_type.iteration_plus_one      3062
20   lr_type.iteration_plus_one      1934
17   lr_type.iteration_plus_one      2904
6    lr_type.iteration_plus_one       235
3           lr_type.sample_size      1362
7    lr_type.iteration_plus_one       531
14   lr_type.iteration_plus_one      2469
4       lr_type.ten_sample_size        74
5   lr_type.hundred_sample_size         2

                                              weights       epsilon  \
19  [[11.472057505397014435], [-4.4349784278962545…  1.000000e-05
11  [[9.087798687760632232], [-4.04964890730847365…  1.000000e-07
12  [[8.305635754643900189], [-4.08145059850403449…  1.000000e-08
22  [[12.136361209371475922], [-4.3072039982668131…  1.000000e-05
9   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
2   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
13  [[7.7814651988257482708], [-4.1149838488482304…  1.000000e-09
10  [[10.283574906574218732], [-4.1060086401810997…  1.000000e-06
23  [[10.550083355026051274], [-4.1522402634849284…  1.000000e-05
18  [[12.378667135942420577], [-5.2467260468350486…  1.000000e-05
21  [[7.574352853182287649], [-4.53548774286644673…  1.000000e-05
1   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
0   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
15  [[10.097329325595773509], [-0.4022719394966720…  1.000000e-05
```

```
8   [[14.944116734687297383], [-5.9888301416859652…  1.000000e-04
16  [[12.0433165615157943], [-3.423846932597623296…  1.000000e-05
20  [[13.077075012640899667], [-5.3331788140696238…  1.000000e-05
17  [[12.144219855790955124], [-5.0777700676349558…  1.000000e-05
6   [[20.268681216307229112], [-17.891141982398493…  1.000000e-02
3   [[3.239870586975073716], [-3.93183866315446415…  1.000000e-05
7   [[18.400300481674902137], [-9.9749477354584466…  1.000000e-03
14  [[1.216749001141672746l], [-3.4149019369519555…  1.000000e-05
4   [[0.07851626733184168337], [-0.114624863637962…  1.000000e-05
5   [[0.0001873121282051282037 8], [-9.962399624765…  1.000000e-05

    elapsed_time is_max_reached        loss  accuracy_kfold       variable
19         1.102          False   85.723087        0.742064            log
11         4.588          False   80.926339        0.739631        epsilon
12         7.677          False   80.216146        0.739354        epsilon
22         0.991          False   85.158078        0.738938            log
9          1.104          False   86.235038        0.738105        epsilon
2          1.110          False   86.235038        0.738105  learning rate
13        15.933          False   79.885937        0.738036        epsilon
10         2.197          False   82.538558        0.737964        epsilon
23         0.867          False   81.190493        0.737131            log
18         1.327          False   83.420795        0.736923            log
21         1.140          False   82.468896        0.735949            log
1          1.660          False   97.099984        0.735186  learning rate
0          5.061          False   97.099984        0.735186   all features
15         0.878          False   83.854123        0.735117            log
8          0.522          False   94.274794        0.734561        epsilon
16         1.237          False   87.890446        0.734283            log
20         0.812          False   85.514273        0.732755            log
17         1.434          False   85.180434        0.730737            log
6          0.151          False  134.658479        0.729835        epsilon
3          0.605          False   80.126535        0.729210  learning rate
7          0.365          False  109.385158        0.728721        epsilon
14         1.117          False   86.710362        0.725452            log
4          0.044          False   98.565776        0.536584  learning rate
5          0.001          False   99.809458        0.535056  learning rate
```

[74]:

```
#combine log Total phenols & Proanthocyanins
ww_x_train_logphenol = log_transform_normalize(ww_x_train_np,5)
ww_x_train_logpp = log_transform_normalize(ww_x_train_logphenol,8)
model_data = run_model('ww','log Total phenols-Proanthocyanins',learning_rate =␣
↪0, learning_rate_type = learning_rate_type.iteration_plus_one ,
                    max_iterations = 150000,epsilon = 1e-5,x_train =␣
↪ww_x_train_logpp , y_train = ww_y_train , model_data = model_data,variable =␣
↪'log')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[74]:
```
   model_name                                   description  \
19         ww                            log Total phenols
24         ww               log Total phenols-Proanthocyanins
11         ww                   whole model-epsilon:1e-07
12         ww                   whole model-epsilon:1e-08
22         ww                         log Proanthocyanins
2          ww   whole model-lr:lr_type.iteration_plus_one
9          ww                   whole model-epsilon:1e-05
13         ww                   whole model-epsilon:1e-09
10         ww                   whole model-epsilon:1e-06
23         ww                                     log Hue
18         ww                               log Magnesium
21         ww                    log Nonflavanoid phenols
1          ww               whole model-lr:lr_type.iteration
0          ww                                 whole model
15         ww                              log Malic acid
8          ww                  whole model-epsilon:0.0001
16         ww                                     log Ash
20         ww                              log Flavanoids
17         ww                       log Alkalinity of ash
6          ww                    whole model-epsilon:0.01
3          ww          whole model-lr:lr_type.sample_size
7          ww                   whole model-epsilon:0.001
14         ww                                 log Alcohol
4          ww      whole model-lr:lr_type.ten_sample_size
5          ww  whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
19  lr_type.iteration_plus_one      2743
24  lr_type.iteration_plus_one      2523
11  lr_type.iteration_plus_one     10464
12  lr_type.iteration_plus_one     19610
22  lr_type.iteration_plus_one      2439
2   lr_type.iteration_plus_one      2753
9   lr_type.iteration_plus_one      2753
13  lr_type.iteration_plus_one     36990
```

```
10   lr_type.iteration_plus_one      5496
23   lr_type.iteration_plus_one      2118
18   lr_type.iteration_plus_one      2205
21   lr_type.iteration_plus_one      1844
1             lr_type.iteration      4032
0             lr_type.iteration      4032
15   lr_type.iteration_plus_one      2115
8    lr_type.iteration_plus_one      1263
16   lr_type.iteration_plus_one      3062
20   lr_type.iteration_plus_one      1934
17   lr_type.iteration_plus_one      2904
6    lr_type.iteration_plus_one       235
3           lr_type.sample_size      1362
7    lr_type.iteration_plus_one       531
14   lr_type.iteration_plus_one      2469
4        lr_type.ten_sample_size        74
5   lr_type.hundred_sample_size         2

                                                 weights       epsilon  \
19  [[11.472057505397014435], [-4.4349784278962545…  1.000000e-05
24  [[11.571783751264503302], [-4.2273367710084355…  1.000000e-05
11  [[9.087798687760632232], [-4.04964890730847365…  1.000000e-07
12  [[8.305635754643900189], [-4.08145059850403449…  1.000000e-08
22  [[12.136361209371475922], [-4.3072039982668131…  1.000000e-05
2   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
9   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
13  [[7.781465198825748270 8], [-4.1149838488482304…  1.000000e-09
10  [[10.283574906574218732], [-4.1060086401810997…  1.000000e-06
23  [[10.550083355026051274], [-4.1522402634849284…  1.000000e-05
18  [[12.378667135942420577], [-5.2467260468350486…  1.000000e-05
21  [[7.574352853182287649], [-4.53548774286644673…  1.000000e-05
1   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
0   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
15  [[10.097329325595773509], [-0.4022719394966720…  1.000000e-05
8   [[14.944116734687297383], [-5.9888301416859652…  1.000000e-04
16  [[12.0433165615157943], [-3.423846932597623296…  1.000000e-05
20  [[13.077075012640899667], [-5.3331788140696238…  1.000000e-05
17  [[12.144219855790955124], [-5.0777700676349558…  1.000000e-05
6   [[20.268681216307229112], [-17.891141982398493…  1.000000e-02
3   [[3.239870586975073716], [-3.93183866315446415…  1.000000e-05
7   [[18.400300481674902137], [-9.9749477354584466…  1.000000e-03
14  [[1.216749001141672746 1], [-3.4149019369519555…  1.000000e-05
4   [[0.0785162673318416833 7], [-0.114624863637962…  1.000000e-05
5   [[0.000187312128205128203 78], [-9.962399624765…  1.000000e-05

   elapsed_time is_max_reached        loss  accuracy_kfold       variable
19        1.102          False   85.723087        0.742064            log
```

```
24          1.572        False   84.807459        0.740256             log
11          4.588        False   80.926339        0.739631         epsilon
12          7.677        False   80.216146        0.739354         epsilon
22          0.991        False   85.158078        0.738938             log
2           1.110        False   86.235038        0.738105   learning rate
9           1.104        False   86.235038        0.738105         epsilon
13         15.933        False   79.885937        0.738036         epsilon
10          2.197        False   82.538558        0.737964         epsilon
23          0.867        False   81.190493        0.737131             log
18          1.327        False   83.420795        0.736923             log
21          1.140        False   82.468896        0.735949             log
1           1.660        False   97.099984        0.735186   learning rate
0           5.061        False   97.099984        0.735186    all features
15          0.878        False   83.854123        0.735117             log
8           0.522        False   94.274794        0.734561         epsilon
16          1.237        False   87.890446        0.734283             log
20          0.812        False   85.514273        0.732755             log
17          1.434        False   85.180434        0.730737             log
6           0.151        False  134.658479        0.729835         epsilon
3           0.605        False   80.126535        0.729210   learning rate
7           0.365        False  109.385158        0.728721         epsilon
14          1.117        False   86.710362        0.725452             log
4           0.044        False   98.565776        0.536584   learning rate
5           0.001        False   99.809458        0.535056   learning rate
```

[75]:
```
show_sorted_model(model_data)
```

[75]:
```
   model_name                                  description  \
19         ww                           log Total phenols
24         ww         log Total phenols-Proanthocyanins
11         ww                 whole model-epsilon:1e-07
12         ww                 whole model-epsilon:1e-08
22         ww                       log Proanthocyanins
2          ww   whole model-lr:lr_type.iteration_plus_one
9          ww                 whole model-epsilon:1e-05
13         ww                 whole model-epsilon:1e-09
10         ww                 whole model-epsilon:1e-06
23         ww                                   log Hue
18         ww                             log Magnesium
21         ww                  log Nonflavanoid phenols
1          ww          whole model-lr:lr_type.iteration
0          ww                               whole model
15         ww                            log Malic acid
8          ww                whole model-epsilon:0.0001
16         ww                                   log Ash
20         ww                            log Flavanoids
17         ww                     log Alkalinity of ash
```

```
6          ww                   whole model-epsilon:0.01
3          ww          whole model-lr:lr_type.sample_size
7          ww                  whole model-epsilon:0.001
14         ww                                log Alcohol
4          ww      whole model-lr:lr_type.ten_sample_size
5          ww  whole model-lr:lr_type.hundred_sample_size

                  learning_rate iteration  \
19   lr_type.iteration_plus_one      2743
24   lr_type.iteration_plus_one      2523
11   lr_type.iteration_plus_one     10464
12   lr_type.iteration_plus_one     19610
22   lr_type.iteration_plus_one      2439
2    lr_type.iteration_plus_one      2753
9    lr_type.iteration_plus_one      2753
13   lr_type.iteration_plus_one     36990
10   lr_type.iteration_plus_one      5496
23   lr_type.iteration_plus_one      2118
18   lr_type.iteration_plus_one      2205
21   lr_type.iteration_plus_one      1844
1             lr_type.iteration      4032
0             lr_type.iteration      4032
15   lr_type.iteration_plus_one      2115
8    lr_type.iteration_plus_one      1263
16   lr_type.iteration_plus_one      3062
20   lr_type.iteration_plus_one      1934
17   lr_type.iteration_plus_one      2904
6    lr_type.iteration_plus_one       235
3           lr_type.sample_size      1362
7    lr_type.iteration_plus_one       531
14   lr_type.iteration_plus_one      2469
4       lr_type.ten_sample_size        74
5   lr_type.hundred_sample_size         2

                                              weights       epsilon  \
19  [[11.472057505397014435], [-4.4349784278962545…  1.000000e-05
24  [[11.571783751264503302], [-4.2273367710084355…  1.000000e-05
11  [[9.087798687760632232], [-4.04964890730847365…  1.000000e-07
12  [[8.305635754643900189], [-4.08145059850403449…  1.000000e-08
22  [[12.136361209371475922], [-4.3072039982668131…  1.000000e-05
2   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
9   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
13  [[7.7814651988257482708], [-4.1149838488482304…  1.000000e-09
10  [[10.283574906574218732], [-4.1060086401810997…  1.000000e-06
23  [[10.550083355026051274], [-4.1522402634849284…  1.000000e-05
18  [[12.378667135942420577], [-5.2467260468350486…  1.000000e-05
21  [[7.574352853182287649], [-4.53548774286644673…  1.000000e-05
```

```
1   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
0   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
15  [[10.097329325595773509], [-0.4022719394966720…  1.000000e-05
8   [[14.944116734687297383], [-5.9888301416859652…  1.000000e-04
16  [[12.0433165615157943], [-3.423846932597623296…  1.000000e-05
20  [[13.077075012640899667], [-5.3331788140696238…  1.000000e-05
17  [[12.144219855790955124], [-5.0777700676349558…  1.000000e-05
6   [[20.268681216307229112], [-17.891141982398493…  1.000000e-02
3   [[3.239870586975073716], [-3.93183866315446415…  1.000000e-05
7   [[18.400300481674902137], [-9.9749477354584466…  1.000000e-03
14  [[1.2167490011416727461], [-3.4149019369519555…  1.000000e-05
4   [[0.07851626733184168337], [-0.114624863637962…  1.000000e-05
5   [[0.00018731212820512820378], [-9.962399624765…  1.000000e-05

    elapsed_time is_max_reached        loss  accuracy_kfold       variable
19         1.102          False   85.723087        0.742064            log
24         1.572          False   84.807459        0.740256            log
11         4.588          False   80.926339        0.739631        epsilon
12         7.677          False   80.216146        0.739354        epsilon
22         0.991          False   85.158078        0.738938            log
2          1.110          False   86.235038        0.738105  learning rate
9          1.104          False   86.235038        0.738105        epsilon
13        15.933          False   79.885937        0.738036        epsilon
10         2.197          False   82.538558        0.737964        epsilon
23         0.867          False   81.190493        0.737131            log
18         1.327          False   83.420795        0.736923            log
21         1.140          False   82.468896        0.735949            log
1          1.660          False   97.099984        0.735186  learning rate
0          5.061          False   97.099984        0.735186   all features
15         0.878          False   83.854123        0.735117            log
8          0.522          False   94.274794        0.734561        epsilon
16         1.237          False   87.890446        0.734283            log
20         0.812          False   85.514273        0.732755            log
17         1.434          False   85.180434        0.730737            log
6          0.151          False  134.658479        0.729835        epsilon
3          0.605          False   80.126535        0.729210  learning rate
7          0.365          False  109.385158        0.728721        epsilon
14         1.117          False   86.710362        0.725452            log
4          0.044          False   98.565776        0.536584  learning rate
5          0.001          False   99.809458        0.535056  learning rate
```

[76]:
```python
#check for removing variables
for i in range(ww_x_train_logpp.shape[1]):
    print (i,',column=>',ww_columns[i])
    ww_x_train_modified = np.delete(ww_x_train_logpp.to_numpy(), [i], 1)
    title = 'log Total phen-Proan-no {}'.format(ww_columns[i])
```

```
    model_data = run_model('ww',title,learning_rate = 0, learning_rate_type =␣
  ↪learning_rate_type.iteration_plus_one ,
                       max_iterations = 150000,epsilon = 1e-5,x_train =␣
  ↪ww_x_train_modified , y_train = ww_y_train , model_data = model_data,
                       variable = 'feature removal')
show_sorted_model(model_data)
```

```
0 ,column=> Alcohol
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
1 ,column=> Malic acid
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
2 ,column=> Ash
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
3 ,column=> Alkalinity of ash
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
```

```
i======> 6
i======> 7
i======> 8
i======> 9
4 ,column=> Magnesium
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
5 ,column=> Total phenols
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
6 ,column=> Flavanoids
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
7 ,column=> Nonflavanoid phenols
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

```
8 ,column=> Proanthocyanins
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
9 ,column=> Hue
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[76]:

```
   model_name                                     description  \
19         ww                               log Total phenols
24         ww               log Total phenols-Proanthocyanins
11         ww                     whole model-epsilon:1e-07
12         ww                     whole model-epsilon:1e-08
22         ww                             log Proanthocyanins
28         ww     log Total phen-Proan-no Alkalinity of ash
2          ww     whole model-lr:lr_type.iteration_plus_one
9          ww                     whole model-epsilon:1e-05
13         ww                     whole model-epsilon:1e-09
10         ww                     whole model-epsilon:1e-06
23         ww                                         log Hue
18         ww                                   log Magnesium
27         ww                     log Total phen-Proan-no Ash
29         ww               log Total phen-Proan-no Magnesium
21         ww                       log Nonflavanoid phenols
1          ww               whole model-lr:lr_type.iteration
0          ww                                     whole model
15         ww                                  log Malic acid
30         ww           log Total phen-Proan-no Total phenols
8          ww                    whole model-epsilon:0.0001
16         ww                                         log Ash
20         ww                                  log Flavanoids
17         ww                         log Alkalinity of ash
```

```
6          ww                        whole model-epsilon:0.01
3          ww              whole model-lr:lr_type.sample_size
7          ww                       whole model-epsilon:0.001
33         ww       log Total phen-Proan-no Proanthocyanins
14         ww                                   log Alcohol
26         ww              log Total phen-Proan-no Malic acid
25         ww                 log Total phen-Proan-no Alcohol
34         ww                     log Total phen-Proan-no Hue
31         ww              log Total phen-Proan-no Flavanoids
32         ww  log Total phen-Proan-no Nonflavanoid phenols
4          ww          whole model-lr:lr_type.ten_sample_size
5          ww    whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
19   lr_type.iteration_plus_one      2743
24   lr_type.iteration_plus_one      2523
11   lr_type.iteration_plus_one     10464
12   lr_type.iteration_plus_one     19610
22   lr_type.iteration_plus_one      2439
28   lr_type.iteration_plus_one      2267
2    lr_type.iteration_plus_one      2753
9    lr_type.iteration_plus_one      2753
13   lr_type.iteration_plus_one     36990
10   lr_type.iteration_plus_one      5496
23   lr_type.iteration_plus_one      2118
18   lr_type.iteration_plus_one      2205
27   lr_type.iteration_plus_one      2961
29   lr_type.iteration_plus_one      2268
21   lr_type.iteration_plus_one      1844
1             lr_type.iteration      4032
0             lr_type.iteration      4032
15   lr_type.iteration_plus_one      2115
30   lr_type.iteration_plus_one      2199
8    lr_type.iteration_plus_one      1263
16   lr_type.iteration_plus_one      3062
20   lr_type.iteration_plus_one      1934
17   lr_type.iteration_plus_one      2904
6    lr_type.iteration_plus_one       235
3           lr_type.sample_size      1362
7    lr_type.iteration_plus_one       531
33   lr_type.iteration_plus_one      2409
14   lr_type.iteration_plus_one      2469
26   lr_type.iteration_plus_one      2721
25   lr_type.iteration_plus_one      2146
34   lr_type.iteration_plus_one      1727
31   lr_type.iteration_plus_one      1994
32   lr_type.iteration_plus_one      1612
```

```
4        lr_type.ten_sample_size        74
5   lr_type.hundred_sample_size         2

                                                weights       epsilon  \
19  [[11.472057505397014435], [-4.4349784278962545…  1.000000e-05
24  [[11.571783751264503302], [-4.2273367710084355…  1.000000e-05
11  [[9.087798687760632232], [-4.04964890730847365…  1.000000e-07
12  [[8.305635754643900189], [-4.08145059850403449…  1.000000e-08
22  [[12.136361209371475922], [-4.3072039982668131…  1.000000e-05
28  [[8.850541831527898764], [-3.89118824863195484…  1.000000e-05
2   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
9   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
13  [[7.781465198825748270 8], [-4.1149838488482304…  1.000000e-09
10  [[10.283574906574218732], [-4.1060086401810997…  1.000000e-06
23  [[10.550083355026051274], [-4.1522402634849284…  1.000000e-05
18  [[12.378667135942420577], [-5.2467260468350486…  1.000000e-05
27  [[11.456849077578187573], [-2.8345242831094986…  1.000000e-05
29  [[14.073646908470692887], [-5.5047098570100916…  1.000000e-05
21  [[7.574352853182287649], [-4.53548774286644673…  1.000000e-05
1   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
0   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
15  [[10.097329325595773509], [-0.4022719394966720…  1.000000e-05
30  [[12.710324024997548719], [-4.8963523723579962…  1.000000e-05
8   [[14.944116734687297383], [-5.9888301416859652…  1.000000e-04
16  [[12.0433165615157943], [-3.423846932597623296…  1.000000e-05
20  [[13.077075012640899667], [-5.3331788140696238…  1.000000e-05
17  [[12.144219855790955124], [-5.0777700676349558…  1.000000e-05
6   [[20.268681216307229112], [-17.891141982398493…  1.000000e-02
3   [[3.239870586975073716], [-3.93183866315446415…  1.000000e-05
7   [[18.400300481674902137], [-9.9749477354584466…  1.000000e-03
33  [[6.573151944209476847], [-3.68075864668547587…  1.000000e-05
14  [[1.216749001141672 7461], [-3.4149019369519555…  1.000000e-05
26  [[9.986283436440546812], [0.720063750748263452…  1.000000e-05
25  [[-2.6197091200274779232], [1.9603738862527356…  1.000000e-05
34  [[9.765787043289230331], [-5.42284373002951372…  1.000000e-05
31  [[15.0500309728559125205], [-5.361661631222361…  1.000000e-05
32  [[-1.0460067470590719306], [-4.565756503472974…  1.000000e-05
4   [[0.0785162673318416833 7], [-0.114624863637962…  1.000000e-05
5   [[0.00018731212820512820378], [-9.962399624765…  1.000000e-05

   elapsed_time is_max_reached       loss  accuracy_kfold         variable
19        1.102          False  85.723087        0.742064              log
24        1.572          False  84.807459        0.740256              log
11        4.588          False  80.926339        0.739631          epsilon
12        7.677          False  80.216146        0.739354          epsilon
22        0.991          False  85.158078        0.738938              log
28        1.235          False  84.164349        0.738384  feature removal
```

```
2          1.110          False    86.235038         0.738105    learning rate
9          1.104          False    86.235038         0.738105          epsilon
13        15.933          False    79.885937         0.738036          epsilon
10         2.197          False    82.538558         0.737964          epsilon
23         0.867          False    81.190493         0.737131              log
18         1.327          False    83.420795         0.736923              log
27         1.427          False    86.799715         0.736504  feature removal
29         0.888          False    83.367648         0.736227  feature removal
21         1.140          False    82.468896         0.735949              log
1          1.660          False    97.099984         0.735186    learning rate
0          5.061          False    97.099984         0.735186     all features
15         0.878          False    83.854123         0.735117              log
30         0.855          False    85.039416         0.734769  feature removal
8          0.522          False    94.274794         0.734561          epsilon
16         1.237          False    87.890446         0.734283              log
20         0.812          False    85.514273         0.732755              log
17         1.434          False    85.180434         0.730737              log
6          0.151          False   134.658479         0.729835          epsilon
3          0.605          False    80.126535         0.729210    learning rate
7          0.365          False   109.385158         0.728721          epsilon
33         1.363          False    84.941279         0.728169  feature removal
14         1.117          False    86.710362         0.725452              log
26         1.080          False    86.886354         0.724344  feature removal
25         0.909          False    85.443068         0.722117  feature removal
34         0.682          False    84.911353         0.721982  feature removal
31         0.825          False    86.027563         0.720732  feature removal
32         0.645          False    86.142293         0.709404  feature removal
4          0.044          False    98.565776         0.536584    learning rate
5          0.001          False    99.809458         0.535056    learning rate
```

[77]:
```
model_data[model_data['variable'] == 'feature removal'].
    ↪sort_values(by=['accuracy_kfold'], ascending=False)
```

[77]:
```
   model_name                                  description  \
28         ww     log Total phen-Proan-no Alkalinity of ash
27         ww                   log Total phen-Proan-no Ash
29         ww             log Total phen-Proan-no Magnesium
30         ww         log Total phen-Proan-no Total phenols
33         ww       log Total phen-Proan-no Proanthocyanins
26         ww            log Total phen-Proan-no Malic acid
25         ww               log Total phen-Proan-no Alcohol
34         ww                   log Total phen-Proan-no Hue
31         ww            log Total phen-Proan-no Flavanoids
32         ww  log Total phen-Proan-no Nonflavanoid phenols

                learning_rate iteration  \
28  lr_type.iteration_plus_one      2267
```

```
27  lr_type.iteration_plus_one      2961
29  lr_type.iteration_plus_one      2268
30  lr_type.iteration_plus_one      2199
33  lr_type.iteration_plus_one      2409
26  lr_type.iteration_plus_one      2721
25  lr_type.iteration_plus_one      2146
34  lr_type.iteration_plus_one      1727
31  lr_type.iteration_plus_one      1994
32  lr_type.iteration_plus_one      1612

                                              weights  epsilon  elapsed_time  \
28  [[8.850541831527898764], [-3.89118824863195484…  0.00001         1.235
27  [[11.456849077578187573], [-2.8345242831094986…  0.00001         1.427
29  [[14.073646908470692887], [-5.5047098570100916…  0.00001         0.888
30  [[12.710324024997548719], [-4.8963523723579962…  0.00001         0.855
33  [[6.573151944209476847], [-3.68075864668547587…  0.00001         1.363
26  [[9.986283436440546812], [0.720063750748263452…  0.00001         1.080
25  [[-2.6197091200274779232], [1.9603738862527356…  0.00001         0.909
34  [[9.765787043289230331], [-5.42284373002951372…  0.00001         0.682
31  [[15.0500309728559125205], [-5.361661631222361…  0.00001         0.825
32  [[-1.0460067470590719306], [-4.565756503472974…  0.00001         0.645

   is_max_reached       loss  accuracy_kfold          variable
28          False  84.164349        0.738384  feature removal
27          False  86.799715        0.736504  feature removal
29          False  83.367648        0.736227  feature removal
30          False  85.039416        0.734769  feature removal
33          False  84.941279        0.728169  feature removal
26          False  86.886354        0.724344  feature removal
25          False  85.443068        0.722117  feature removal
34          False  84.911353        0.721982  feature removal
31          False  86.027563        0.720732  feature removal
32          False  86.142293        0.709404  feature removal
```

[78]:
```
show_sorted_model(model_data)
```

[78]:
```
   model_name                                   description  \
19         ww                             log Total phenols
24         ww             log Total phenols-Proanthocyanins
11         ww                    whole model-epsilon:1e-07
12         ww                    whole model-epsilon:1e-08
22         ww                         log Proanthocyanins
28         ww     log Total phen-Proan-no Alkalinity of ash
2          ww     whole model-lr:lr_type.iteration_plus_one
9          ww                    whole model-epsilon:1e-05
13         ww                    whole model-epsilon:1e-09
10         ww                    whole model-epsilon:1e-06
```

23	ww	log Hue
18	ww	log Magnesium
27	ww	log Total phen-Proan-no Ash
29	ww	log Total phen-Proan-no Magnesium
21	ww	log Nonflavanoid phenols
1	ww	whole model-lr:lr_type.iteration
0	ww	whole model
15	ww	log Malic acid
30	ww	log Total phen-Proan-no Total phenols
8	ww	whole model-epsilon:0.0001
16	ww	log Ash
20	ww	log Flavanoids
17	ww	log Alkalinity of ash
6	ww	whole model-epsilon:0.01
3	ww	whole model-lr:lr_type.sample_size
7	ww	whole model-epsilon:0.001
33	ww	log Total phen-Proan-no Proanthocyanins
14	ww	log Alcohol
26	ww	log Total phen-Proan-no Malic acid
25	ww	log Total phen-Proan-no Alcohol
34	ww	log Total phen-Proan-no Hue
31	ww	log Total phen-Proan-no Flavanoids
32	ww	log Total phen-Proan-no Nonflavanoid phenols
4	ww	whole model-lr:lr_type.ten_sample_size
5	ww	whole model-lr:lr_type.hundred_sample_size

	learning_rate	iteration	\
19	lr_type.iteration_plus_one	2743	
24	lr_type.iteration_plus_one	2523	
11	lr_type.iteration_plus_one	10464	
12	lr_type.iteration_plus_one	19610	
22	lr_type.iteration_plus_one	2439	
28	lr_type.iteration_plus_one	2267	
2	lr_type.iteration_plus_one	2753	
9	lr_type.iteration_plus_one	2753	
13	lr_type.iteration_plus_one	36990	
10	lr_type.iteration_plus_one	5496	
23	lr_type.iteration_plus_one	2118	
18	lr_type.iteration_plus_one	2205	
27	lr_type.iteration_plus_one	2961	
29	lr_type.iteration_plus_one	2268	
21	lr_type.iteration_plus_one	1844	
1	lr_type.iteration	4032	
0	lr_type.iteration	4032	
15	lr_type.iteration_plus_one	2115	
30	lr_type.iteration_plus_one	2199	
8	lr_type.iteration_plus_one	1263	

```
16   lr_type.iteration_plus_one      3062
20   lr_type.iteration_plus_one      1934
17   lr_type.iteration_plus_one      2904
6    lr_type.iteration_plus_one       235
3           lr_type.sample_size      1362
7    lr_type.iteration_plus_one       531
33   lr_type.iteration_plus_one      2409
14   lr_type.iteration_plus_one      2469
26   lr_type.iteration_plus_one      2721
25   lr_type.iteration_plus_one      2146
34   lr_type.iteration_plus_one      1727
31   lr_type.iteration_plus_one      1994
32   lr_type.iteration_plus_one      1612
4        lr_type.ten_sample_size        74
5    lr_type.hundred_sample_size         2

                                              weights       epsilon  \
19  [[11.472057505397014435], [-4.4349784278962545…  1.000000e-05
24  [[11.571783751264503302], [-4.2273367710084355…  1.000000e-05
11  [[9.087798687760632232], [-4.04964890730847365…  1.000000e-07
12  [[8.305635754643900189], [-4.08145059850403449…  1.000000e-08
22  [[12.136361209371475922], [-4.3072039982668131…  1.000000e-05
28  [[8.850541831527898764], [-3.89118824863195484…  1.000000e-05
2   [[12.137409248966954996 4], [-4.520518798900213…  1.000000e-05
9   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
13  [[7.781465198825748 2708], [-4.1149838488482304…  1.000000e-09
10  [[10.283574906574218732], [-4.1060086401810997…  1.000000e-06
23  [[10.550083355026051274], [-4.1522402634849284…  1.000000e-05
18  [[12.378667135942420577], [-5.2467260468350486…  1.000000e-05
27  [[11.456849077578187573], [-2.8345242831094986…  1.000000e-05
29  [[14.073646908470692887], [-5.5047098570100916…  1.000000e-05
21  [[7.574352853182287649], [-4.53548774286644673…  1.000000e-05
1   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
0   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
15  [[10.097329325595773509], [-0.4022719394966720…  1.000000e-05
30  [[12.710324024997548719], [-4.8963523723579962…  1.000000e-05
8   [[14.944116734687297383], [-5.9888301416859652…  1.000000e-04
16  [[12.0433165615157943], [-3.423846932597623296…  1.000000e-05
20  [[13.077075012640899667], [-5.3331788140696238…  1.000000e-05
17  [[12.144219855790955124], [-5.0777700676349558…  1.000000e-05
6   [[20.268681216307229112], [-17.891141982398493…  1.000000e-02
3   [[3.239870586975073716], [-3.93183866315446415…  1.000000e-05
7   [[18.400300481674902137], [-9.9749477354584466…  1.000000e-03
33  [[6.573151944209476847], [-3.68075864668547587…  1.000000e-05
14  [[1.216749001141672746 1], [-3.4149019369519555…  1.000000e-05
26  [[9.986283436440546812], [0.720063750748263452…  1.000000e-05
25  [[-2.6197091200274779232], [1.9603738862527356…  1.000000e-05
```

```
34  [[9.765787043289230331], [-5.42284373002951372…  1.000000e-05
31  [[15.050030972855912520 5], [-5.361661631222361…  1.000000e-05
32  [[-1.0460067470590719306], [-4.565756503472974…  1.000000e-05
4   [[0.07851626733184168337], [-0.114624863637962…  1.000000e-05
5   [[0.00018731212820512820378], [-9.962399624765…  1.000000e-05
```

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
19	1.102	False	85.723087	0.742064	log
24	1.572	False	84.807459	0.740256	log
11	4.588	False	80.926339	0.739631	epsilon
12	7.677	False	80.216146	0.739354	epsilon
22	0.991	False	85.158078	0.738938	log
28	1.235	False	84.164349	0.738384	feature removal
2	1.110	False	86.235038	0.738105	learning rate
9	1.104	False	86.235038	0.738105	epsilon
13	15.933	False	79.885937	0.738036	epsilon
10	2.197	False	82.538558	0.737964	epsilon
23	0.867	False	81.190493	0.737131	log
18	1.327	False	83.420795	0.736923	log
27	1.427	False	86.799715	0.736504	feature removal
29	0.888	False	83.367648	0.736227	feature removal
21	1.140	False	82.468896	0.735949	log
1	1.660	False	97.099984	0.735186	learning rate
0	5.061	False	97.099984	0.735186	all features
15	0.878	False	83.854123	0.735117	log
30	0.855	False	85.039416	0.734769	feature removal
8	0.522	False	94.274794	0.734561	epsilon
16	1.237	False	87.890446	0.734283	log
20	0.812	False	85.514273	0.732755	log
17	1.434	False	85.180434	0.730737	log
6	0.151	False	134.658479	0.729835	epsilon
3	0.605	False	80.126535	0.729210	learning rate
7	0.365	False	109.385158	0.728721	epsilon
33	1.363	False	84.941279	0.728169	feature removal
14	1.117	False	86.710362	0.725452	log
26	1.080	False	86.886354	0.724344	feature removal
25	0.909	False	85.443068	0.722117	feature removal
34	0.682	False	84.911353	0.721982	feature removal
31	0.825	False	86.027563	0.720732	feature removal
32	0.645	False	86.142293	0.709404	feature removal
4	0.044	False	98.565776	0.536584	learning rate
5	0.001	False	99.809458	0.535056	learning rate

[79]:
```python
ww_x_finallog = pd.DataFrame(ww_x_train_logpp)
```

[80]:
```python
#test power 2 of features
for i in range(ww_x_finallog.shape[1]):
```

```
    print (i,',column=>',ww_columns[i])
    x_train_power = power_n_feature(ww_x_finallog,i,2).to_numpy()
    title = 'log Total phen-Proan-*2 {}'.format(ww_columns[i])
    print(title)
    model_data = run_model('ww',title,learning_rate = 0, learning_rate_type =␣
↪learning_rate_type.iteration_plus_one ,max_iterations = 150000,
                          epsilon = 1e-5,x_train = x_train_power , y_train =␣
↪ww_y_train , model_data = model_data,variable = 'power*2')
show_sorted_model(model_data)
```

```
0 ,column=> Alcohol
log Total phen-Proan-*2 Alcohol
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
1 ,column=> Malic acid
log Total phen-Proan-*2 Malic acid
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
2 ,column=> Ash
log Total phen-Proan-*2 Ash
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

```
3 ,column=> Alkalinity of ash
log Total phen-Proan-*2 Alkalinity of ash
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
4 ,column=> Magnesium
log Total phen-Proan-*2 Magnesium
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
5 ,column=> Total phenols
log Total phen-Proan-*2 Total phenols
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
6 ,column=> Flavanoids
log Total phen-Proan-*2 Flavanoids
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

```
7 ,column=> Nonflavanoid phenols
log Total phen-Proan-*2 Nonflavanoid phenols
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
8 ,column=> Proanthocyanins
log Total phen-Proan-*2 Proanthocyanins
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
9 ,column=> Hue
log Total phen-Proan-*2 Hue
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[80]:

```
   model_name                                 description  \
19         ww                          log Total phenols
38         ww     log Total phen-Proan-*2 Alkalinity of ash
39         ww             log Total phen-Proan-*2 Magnesium
24         ww             log Total phenols-Proanthocyanins
40         ww         log Total phen-Proan-*2 Total phenols
11         ww                   whole model-epsilon:1e-07
12         ww                   whole model-epsilon:1e-08
41         ww            log Total phen-Proan-*2 Flavanoids
22         ww                        log Proanthocyanins
```

```
28         ww     log Total phen-Proan-no Alkalinity of ash
9          ww                      whole model-epsilon:1e-05
2          ww     whole model-lr:lr_type.iteration_plus_one
13         ww                      whole model-epsilon:1e-09
10         ww                      whole model-epsilon:1e-06
23         ww                                        log Hue
18         ww                                  log Magnesium
27         ww                    log Total phen-Proan-no Ash
29         ww              log Total phen-Proan-no Magnesium
36         ww             log Total phen-Proan-*2 Malic acid
21         ww                       log Nonflavanoid phenols
35         ww                log Total phen-Proan-*2 Alcohol
37         ww                    log Total phen-Proan-*2 Ash
0          ww                                    whole model
1          ww               whole model-lr:lr_type.iteration
15         ww                                 log Malic acid
30         ww         log Total phen-Proan-no Total phenols
8          ww                     whole model-epsilon:0.0001
16         ww                                        log Ash
42         ww  log Total phen-Proan-*2 Nonflavanoid phenols
20         ww                                 log Flavanoids
17         ww                          log Alkalinity of ash
6          ww                       whole model-epsilon:0.01
3          ww             whole model-lr:lr_type.sample_size
7          ww                      whole model-epsilon:0.001
33         ww       log Total phen-Proan-no Proanthocyanins
44         ww                    log Total phen-Proan-*2 Hue
43         ww       log Total phen-Proan-*2 Proanthocyanins
14         ww                                    log Alcohol
26         ww             log Total phen-Proan-no Malic acid
25         ww                log Total phen-Proan-no Alcohol
34         ww                    log Total phen-Proan-no Hue
31         ww             log Total phen-Proan-no Flavanoids
32         ww  log Total phen-Proan-no Nonflavanoid phenols
4          ww         whole model-lr:lr_type.ten_sample_size
5          ww     whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
19   lr_type.iteration_plus_one      2743
38   lr_type.iteration_plus_one      2318
39   lr_type.iteration_plus_one      2387
24   lr_type.iteration_plus_one      2523
40   lr_type.iteration_plus_one      3546
11   lr_type.iteration_plus_one     10464
12   lr_type.iteration_plus_one     19610
41   lr_type.iteration_plus_one      2133
22   lr_type.iteration_plus_one      2439
```

```
28   lr_type.iteration_plus_one      2267
9    lr_type.iteration_plus_one      2753
2    lr_type.iteration_plus_one      2753
13   lr_type.iteration_plus_one     36990
10   lr_type.iteration_plus_one      5496
23   lr_type.iteration_plus_one      2118
18   lr_type.iteration_plus_one      2205
27   lr_type.iteration_plus_one      2961
29   lr_type.iteration_plus_one      2268
36   lr_type.iteration_plus_one      2722
21   lr_type.iteration_plus_one      1844
35   lr_type.iteration_plus_one      2334
37   lr_type.iteration_plus_one      2738
0              lr_type.iteration      4032
1              lr_type.iteration      4032
15   lr_type.iteration_plus_one      2115
30   lr_type.iteration_plus_one      2199
8    lr_type.iteration_plus_one      1263
16   lr_type.iteration_plus_one      3062
42   lr_type.iteration_plus_one      2058
20   lr_type.iteration_plus_one      1934
17   lr_type.iteration_plus_one      2904
6    lr_type.iteration_plus_one       235
3            lr_type.sample_size      1362
7    lr_type.iteration_plus_one       531
33   lr_type.iteration_plus_one      2409
44   lr_type.iteration_plus_one      1946
43   lr_type.iteration_plus_one      3611
14   lr_type.iteration_plus_one      2469
26   lr_type.iteration_plus_one      2721
25   lr_type.iteration_plus_one      2146
34   lr_type.iteration_plus_one      1727
31   lr_type.iteration_plus_one      1994
32   lr_type.iteration_plus_one      1612
4        lr_type.ten_sample_size        74
5    lr_type.hundred_sample_size         2

                                              weights       epsilon  \
19  [[11.472057505397014435], [-4.4349784278962545…  1.000000e-05
38  [[10.780086339913878238], [-3.7216570283533134…  1.000000e-05
39  [[11.915738960490262956], [-4.1662803858045918…  1.000000e-05
24  [[11.571783751264503302], [-4.2273367710084355…  1.000000e-05
40  [[16.757890297389447553], [-6.5489032844729216…  1.000000e-05
11  [[9.087798687760632232], [-4.04964890730847365…  1.000000e-07
12  [[8.305635754643900189], [-4.08145059850403449…  1.000000e-08
41  [[12.931171030702525031], [-4.6407389412016888…  1.000000e-05
22  [[12.136361209371475922], [-4.3072039982668131…  1.000000e-05
```

28	[[8.850541831527898764], [-3.89118824863195484…	1.000000e-05
9	[[12.1374092489669549964], [-4.520518798900213…	1.000000e-05
2	[[12.1374092489669549964], [-4.520518798900213…	1.000000e-05
13	[[7.781465198825748270 8], [-4.1149838488482304…	1.000000e-09
10	[[10.283574906574218732], [-4.1060086401810997…	1.000000e-06
23	[[10.550083355026051274], [-4.1522402634849284…	1.000000e-05
18	[[12.378667135942420577], [-5.2467260468350486…	1.000000e-05
27	[[11.456849077578187573], [-2.8345242831094986…	1.000000e-05
29	[[14.073646908470692887], [-5.5047098570100916…	1.000000e-05
36	[[11.727506805508466206], [-8.7890367450832210…	1.000000e-05
21	[[7.574352853182287649], [-4.53548774286644673…	1.000000e-05
35	[[7.99809541520755354], [-3.900010977053474500…	1.000000e-05
37	[[11.810775725772269145], [-3.8200989106924167…	1.000000e-05
0	[[16.378906342486571375], [-5.5259574517590583…	1.000000e-05
1	[[16.378906342486571375], [-5.5259574517590583…	1.000000e-05
15	[[10.097329325595773509], [-0.4022719394966720…	1.000000e-05
30	[[12.710324024997548719], [-4.8963523723579962…	1.000000e-05
8	[[14.944116734687297383], [-5.9888301416859652…	1.000000e-04
16	[[12.0433165615157943], [-3.423846932597623296…	1.000000e-05
42	[[9.578820849453080831], [-4.04663715600940727…	1.000000e-05
20	[[13.077075012640899667], [-5.3331788140696238…	1.000000e-05
17	[[12.144219855790955124], [-5.0777700676349558…	1.000000e-05
6	[[20.268681216307229112], [-17.891141982398493…	1.000000e-02
3	[[3.239870586975073716], [-3.93183866315446415…	1.000000e-05
7	[[18.400300481674902137], [-9.9749477354584466…	1.000000e-03
33	[[6.573151944209476847], [-3.68075864668547587…	1.000000e-05
44	[[11.0619179243542326545], [-4.980700263316878…	1.000000e-05
43	[[9.70288456997957947], [-4.517980142783009670…	1.000000e-05
14	[[1.216749001141672 7461], [-3.4149019369519555…	1.000000e-05
26	[[9.986283436440546812], [0.720063750748263452…	1.000000e-05
25	[[-2.6197091200274779232], [1.9603738862527356…	1.000000e-05
34	[[9.765787043289230331], [-5.42284373002951372…	1.000000e-05
31	[[15.0500309728559125205], [-5.361661631222361…	1.000000e-05
32	[[-1.0460067470590719306], [-4.565756503472974…	1.000000e-05
4	[[0.07851626733184168337], [-0.114624863637962…	1.000000e-05
5	[[0.00018731212820512820378], [-9.962399624765…	1.000000e-05

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
19	1.102	False	85.723087	0.742064	log
38	0.951	False	85.247168	0.741300	power*2
39	0.980	False	84.223758	0.741162	power*2
24	1.572	False	84.807459	0.740256	log
40	1.775	False	92.024182	0.740187	power*2
11	4.588	False	80.926339	0.739631	epsilon
12	7.677	False	80.216146	0.739354	epsilon
41	0.883	False	84.169551	0.739354	power*2
22	0.991	False	85.158078	0.738938	log

28	1.235	False	84.164349	0.738384	feature removal
9	1.104	False	86.235038	0.738105	epsilon
2	1.110	False	86.235038	0.738105	learning rate
13	15.933	False	79.885937	0.738036	epsilon
10	2.197	False	82.538558	0.737964	epsilon
23	0.867	False	81.190493	0.737131	log
18	1.327	False	83.420795	0.736923	log
27	1.427	False	86.799715	0.736504	feature removal
29	0.888	False	83.367648	0.736227	feature removal
36	1.425	False	85.761598	0.736157	power*2
21	1.140	False	82.468896	0.735949	log
35	1.016	False	84.943099	0.735532	power*2
37	1.107	False	85.890549	0.735323	power*2
0	5.061	False	97.099984	0.735186	all features
1	1.660	False	97.099984	0.735186	learning rate
15	0.878	False	83.854123	0.735117	log
30	0.855	False	85.039416	0.734769	feature removal
8	0.522	False	94.274794	0.734561	epsilon
16	1.237	False	87.890446	0.734283	log
42	0.833	False	84.638295	0.733376	power*2
20	0.812	False	85.514273	0.732755	log
17	1.434	False	85.180434	0.730737	log
6	0.151	False	134.658479	0.729835	epsilon
3	0.605	False	80.126535	0.729210	learning rate
7	0.365	False	109.385158	0.728721	epsilon
33	1.363	False	84.941279	0.728169	feature removal
44	0.919	False	85.532771	0.727539	power*2
43	1.697	False	91.446068	0.726500	power*2
14	1.117	False	86.710362	0.725452	log
26	1.080	False	86.886354	0.724344	feature removal
25	0.909	False	85.443068	0.722117	feature removal
34	0.682	False	84.911353	0.721982	feature removal
31	0.825	False	86.027563	0.720732	feature removal
32	0.645	False	86.142293	0.709404	feature removal
4	0.044	False	98.565776	0.536584	learning rate
5	0.001	False	99.809458	0.535056	learning rate

[81]:
```
#alkalanity of ash selected as power2
x_train_power2 = power_n_feature(ww_x_finallog,3,2).to_numpy()
model_data = run_model('ww','log Total phen/Proan-*2 Alkalinity of␣
↪ash',learning_rate = 0, learning_rate_type = learning_rate_type.
↪iteration_plus_one ,max_iterations = 150000,
                        epsilon = 1e-5,x_train = x_train_power2 , y_train =␣
↪ww_y_train , model_data = model_data,variable = 'power*2')
show_sorted_model(model_data)
```

```
i======> 0
```

```
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[81]:

```
   model_name                                     description  \
19         ww                               log Total phenols
45         ww      log Total phen/Proan-*2 Alkalinity of ash
38         ww      log Total phen-Proan-*2 Alkalinity of ash
39         ww              log Total phen-Proan-*2 Magnesium
24         ww              log Total phenols-Proanthocyanins
40         ww          log Total phen-Proan-*2 Total phenols
11         ww                     whole model-epsilon:1e-07
12         ww                     whole model-epsilon:1e-08
41         ww             log Total phen-Proan-*2 Flavanoids
22         ww                         log Proanthocyanins
28         ww      log Total phen-Proan-no Alkalinity of ash
2          ww      whole model-lr:lr_type.iteration_plus_one
9          ww                     whole model-epsilon:1e-05
13         ww                     whole model-epsilon:1e-09
10         ww                     whole model-epsilon:1e-06
23         ww                                     log Hue
18         ww                               log Magnesium
27         ww                    log Total phen-Proan-no Ash
29         ww              log Total phen-Proan-no Magnesium
36         ww             log Total phen-Proan-*2 Malic acid
21         ww                     log Nonflavanoid phenols
35         ww                log Total phen-Proan-*2 Alcohol
37         ww                    log Total phen-Proan-*2 Ash
0          ww                                 whole model
1          ww               whole model-lr:lr_type.iteration
15         ww                              log Malic acid
30         ww          log Total phen-Proan-no Total phenols
8          ww                    whole model-epsilon:0.0001
16         ww                                     log Ash
42         ww   log Total phen-Proan-*2 Nonflavanoid phenols
20         ww                              log Flavanoids
17         ww                      log Alkalinity of ash
6          ww                      whole model-epsilon:0.01
3          ww             whole model-lr:lr_type.sample_size
7          ww                     whole model-epsilon:0.001
33         ww        log Total phen-Proan-no Proanthocyanins
```

```
44         ww                  log Total phen-Proan-*2 Hue
43         ww       log Total phen-Proan-*2 Proanthocyanins
14         ww                                  log Alcohol
26         ww            log Total phen-Proan-no Malic acid
25         ww               log Total phen-Proan-no Alcohol
34         ww                   log Total phen-Proan-no Hue
31         ww            log Total phen-Proan-no Flavanoids
32         ww  log Total phen-Proan-no Nonflavanoid phenols
4          ww        whole model-lr:lr_type.ten_sample_size
5          ww    whole model-lr:lr_type.hundred_sample_size

                   learning_rate iteration  \
19   lr_type.iteration_plus_one      2743
45   lr_type.iteration_plus_one      2318
38   lr_type.iteration_plus_one      2318
39   lr_type.iteration_plus_one      2387
24   lr_type.iteration_plus_one      2523
40   lr_type.iteration_plus_one      3546
11   lr_type.iteration_plus_one     10464
12   lr_type.iteration_plus_one     19610
41   lr_type.iteration_plus_one      2133
22   lr_type.iteration_plus_one      2439
28   lr_type.iteration_plus_one      2267
2    lr_type.iteration_plus_one      2753
9    lr_type.iteration_plus_one      2753
13   lr_type.iteration_plus_one     36990
10   lr_type.iteration_plus_one      5496
23   lr_type.iteration_plus_one      2118
18   lr_type.iteration_plus_one      2205
27   lr_type.iteration_plus_one      2961
29   lr_type.iteration_plus_one      2268
36   lr_type.iteration_plus_one      2722
21   lr_type.iteration_plus_one      1844
35   lr_type.iteration_plus_one      2334
37   lr_type.iteration_plus_one      2738
0             lr_type.iteration      4032
1             lr_type.iteration      4032
15   lr_type.iteration_plus_one      2115
30   lr_type.iteration_plus_one      2199
8    lr_type.iteration_plus_one      1263
16   lr_type.iteration_plus_one      3062
42   lr_type.iteration_plus_one      2058
20   lr_type.iteration_plus_one      1934
17   lr_type.iteration_plus_one      2904
6    lr_type.iteration_plus_one       235
3           lr_type.sample_size      1362
7    lr_type.iteration_plus_one       531
```

```
33   lr_type.iteration_plus_one      2409
44   lr_type.iteration_plus_one      1946
43   lr_type.iteration_plus_one      3611
14   lr_type.iteration_plus_one      2469
26   lr_type.iteration_plus_one      2721
25   lr_type.iteration_plus_one      2146
34   lr_type.iteration_plus_one      1727
31   lr_type.iteration_plus_one      1994
32   lr_type.iteration_plus_one      1612
4       lr_type.ten_sample_size        74
5   lr_type.hundred_sample_size         2

                                                weights       epsilon  \
19  [[11.472057505397014435], [-4.4349784278962545…  1.000000e-05
45  [[10.780086339913878238], [-3.7216570283533134…  1.000000e-05
38  [[10.780086339913878238], [-3.7216570283533134…  1.000000e-05
39  [[11.915738960490262956], [-4.1662803858045918…  1.000000e-05
24  [[11.571783751264503302], [-4.2273367710084355…  1.000000e-05
40  [[16.757890297389447553], [-6.5489032844729216…  1.000000e-05
11  [[9.087798687760632232], [-4.04964890730847365…  1.000000e-07
12  [[8.305635754643900189], [-4.08145059850403449…  1.000000e-08
41  [[12.931171030702525031], [-4.6407389412016888…  1.000000e-05
22  [[12.136361209371475922], [-4.3072039982668131…  1.000000e-05
28  [[8.850541831527898764], [-3.89118824863195484…  1.000000e-05
2   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
9   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
13  [[7.781465198825748270 8], [-4.1149838488482304…  1.000000e-09
10  [[10.283574906574218732], [-4.1060086401810997…  1.000000e-06
23  [[10.550083355026051274], [-4.1522402634849284…  1.000000e-05
18  [[12.378667135942420577], [-5.2467260468350486…  1.000000e-05
27  [[11.456849077578187573], [-2.8345242831094986…  1.000000e-05
29  [[14.073646908470692887], [-5.5047098570100916…  1.000000e-05
36  [[11.727506805508466206], [-8.7890367450832210…  1.000000e-05
21  [[7.574352853182287649], [-4.53548774286644673…  1.000000e-05
35  [[7.99809541520755354], [-3.900010977053474500…  1.000000e-05
37  [[11.810775725772269145], [-3.8200989106924167…  1.000000e-05
0   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
1   [[16.378906342486571375], [-5.5259574517590583…  1.000000e-05
15  [[10.097329325595773509], [-0.4022719394966720…  1.000000e-05
30  [[12.710324024997548719], [-4.8963523723579962…  1.000000e-05
8   [[14.944116734687297383], [-5.9888301416859652…  1.000000e-04
16  [[12.0433165615157943], [-3.423846932597623296…  1.000000e-05
42  [[9.578820849453080831], [-4.04663715600940727…  1.000000e-05
20  [[13.077075012640899667], [-5.3331788140696238…  1.000000e-05
17  [[12.144219855790955124], [-5.0777700676349558…  1.000000e-05
6   [[20.268681216307229112], [-17.891141982398493…  1.000000e-02
3   [[3.239870586975073716], [-3.93183866315446415…  1.000000e-05
```

7	[[18.400300481674902137], [-9.9749477354584466…	1.000000e-03
33	[[6.573151944209476847], [-3.68075864668547587…	1.000000e-05
44	[[11.0619179243542326545], [-4.980700263316878…	1.000000e-05
43	[[9.70288456997957947], [-4.51798014278300967O…	1.000000e-05
14	[[1.216749001141672746 1], [-3.4149019369519555…	1.000000e-05
26	[[9.986283436440546812], [0.720063750748263452…	1.000000e-05
25	[[-2.6197091200274779232], [1.9603738862527356…	1.000000e-05
34	[[9.765787043289230331], [-5.42284373002951372…	1.000000e-05
31	[[15.0500309728559125205], [-5.361661631222361…	1.000000e-05
32	[[-1.0460067470590719306], [-4.565756503472974…	1.000000e-05
4	[[0.07851626733184168337], [-0.114624863637962…	1.000000e-05
5	[[0.00018731212820512820378], [-9.962399624765…	1.000000e-05

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
19	1.102	False	85.723087	0.742064	log
45	0.991	False	85.247168	0.741300	power*2
38	0.951	False	85.247168	0.741300	power*2
39	0.980	False	84.223758	0.741162	power*2
24	1.572	False	84.807459	0.740256	log
40	1.775	False	92.024182	0.740187	power*2
11	4.588	False	80.926339	0.739631	epsilon
12	7.677	False	80.216146	0.739354	epsilon
41	0.883	False	84.169551	0.739354	power*2
22	0.991	False	85.158078	0.738938	log
28	1.235	False	84.164349	0.738384	feature removal
2	1.110	False	86.235038	0.738105	learning rate
9	1.104	False	86.235038	0.738105	epsilon
13	15.933	False	79.885937	0.738036	epsilon
10	2.197	False	82.538558	0.737964	epsilon
23	0.867	False	81.190493	0.737131	log
18	1.327	False	83.420795	0.736923	log
27	1.427	False	86.799715	0.736504	feature removal
29	0.888	False	83.367648	0.736227	feature removal
36	1.425	False	85.761598	0.736157	power*2
21	1.140	False	82.468896	0.735949	log
35	1.016	False	84.943099	0.735532	power*2
37	1.107	False	85.890549	0.735323	power*2
0	5.061	False	97.099984	0.735186	all features
1	1.660	False	97.099984	0.735186	learning rate
15	0.878	False	83.854123	0.735117	log
30	0.855	False	85.039416	0.734769	feature removal
8	0.522	False	94.274794	0.734561	epsilon
16	1.237	False	87.890446	0.734283	log
42	0.833	False	84.638295	0.733376	power*2
20	0.812	False	85.514273	0.732755	log
17	1.434	False	85.180434	0.730737	log
6	0.151	False	134.658479	0.729835	epsilon

```
3           0.605         False    80.126535        0.729210    learning rate
7           0.365         False   109.385158        0.728721          epsilon
33          1.363         False    84.941279        0.728169  feature removal
44          0.919         False    85.532771        0.727539          power*2
43          1.697         False    91.446068        0.726500          power*2
14          1.117         False    86.710362        0.725452              log
26          1.080         False    86.886354        0.724344  feature removal
25          0.909         False    85.443068        0.722117  feature removal
34          0.682         False    84.911353        0.721982  feature removal
31          0.825         False    86.027563        0.720732  feature removal
32          0.645         False    86.142293        0.709404  feature removal
4           0.044         False    98.565776        0.536584    learning rate
5           0.001         False    99.809458        0.535056    learning rate
```

[82]:

```
#alkalanity of ash-magnaseim as power2 test
x_train_power2 = power_n_feature(ww_x_finallog,3,2).to_numpy()
x_train_powerashmag = power_n_feature(pd.DataFrame(x_train_power2),4,2).
 ↪to_numpy()
model_data = run_model('ww','log Total phen/Proan-*2 Alkal of ash/
 ↪mage',learning_rate = 0, learning_rate_type = learning_rate_type.
 ↪iteration_plus_one ,max_iterations = 150000,
                       epsilon = 1e-5,x_train = x_train_powerashmag , y_train =␣
 ↪ww_y_train , model_data = model_data,variable = 'power*2')
show_sorted_model(model_data)
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[82]:

```
   model_name                                   description  \
19         ww                            log Total phenols
46         ww     log Total phen/Proan-*2 Alkal of ash/mage
45         ww     log Total phen/Proan-*2 Alkalinity of ash
38         ww     log Total phen-Proan-*2 Alkalinity of ash
39         ww             log Total phen-Proan-*2 Magnesium
24         ww             log Total phenols-Proanthocyanins
40         ww        log Total phen-Proan-*2 Total phenols
11         ww                  whole model-epsilon:1e-07
12         ww                  whole model-epsilon:1e-08
41         ww            log Total phen-Proan-*2 Flavanoids
```

```
22         ww                             log Proanthocyanins
28         ww    log Total phen-Proan-no Alkalinity of ash
2          ww    whole model-lr:lr_type.iteration_plus_one
9          ww                  whole model-epsilon:1e-05
13         ww                  whole model-epsilon:1e-09
10         ww                  whole model-epsilon:1e-06
23         ww                                    log Hue
18         ww                              log Magnesium
27         ww                 log Total phen-Proan-no Ash
29         ww           log Total phen-Proan-no Magnesium
36         ww          log Total phen-Proan-*2 Malic acid
21         ww                  log Nonflavanoid phenols
35         ww             log Total phen-Proan-*2 Alcohol
37         ww                 log Total phen-Proan-*2 Ash
0          ww                                whole model
1          ww            whole model-lr:lr_type.iteration
15         ww                             log Malic acid
30         ww        log Total phen-Proan-no Total phenols
8          ww                 whole model-epsilon:0.0001
16         ww                                    log Ash
42         ww  log Total phen-Proan-*2 Nonflavanoid phenols
20         ww                             log Flavanoids
17         ww                       log Alkalinity of ash
6          ww                   whole model-epsilon:0.01
3          ww          whole model-lr:lr_type.sample_size
7          ww                  whole model-epsilon:0.001
33         ww       log Total phen-Proan-no Proanthocyanins
44         ww                 log Total phen-Proan-*2 Hue
43         ww       log Total phen-Proan-*2 Proanthocyanins
14         ww                                log Alcohol
26         ww          log Total phen-Proan-no Malic acid
25         ww             log Total phen-Proan-no Alcohol
34         ww                 log Total phen-Proan-no Hue
31         ww          log Total phen-Proan-no Flavanoids
32         ww  log Total phen-Proan-no Nonflavanoid phenols
4          ww       whole model-lr:lr_type.ten_sample_size
5          ww   whole model-lr:lr_type.hundred_sample_size

                 learning_rate iteration  \
19   lr_type.iteration_plus_one      2743
46   lr_type.iteration_plus_one      2189
45   lr_type.iteration_plus_one      2318
38   lr_type.iteration_plus_one      2318
39   lr_type.iteration_plus_one      2387
24   lr_type.iteration_plus_one      2523
40   lr_type.iteration_plus_one      3546
11   lr_type.iteration_plus_one     10464
```

```
12   lr_type.iteration_plus_one     19610
41   lr_type.iteration_plus_one      2133
22   lr_type.iteration_plus_one      2439
28   lr_type.iteration_plus_one      2267
2    lr_type.iteration_plus_one      2753
9    lr_type.iteration_plus_one      2753
13   lr_type.iteration_plus_one     36990
10   lr_type.iteration_plus_one      5496
23   lr_type.iteration_plus_one      2118
18   lr_type.iteration_plus_one      2205
27   lr_type.iteration_plus_one      2961
29   lr_type.iteration_plus_one      2268
36   lr_type.iteration_plus_one      2722
21   lr_type.iteration_plus_one      1844
35   lr_type.iteration_plus_one      2334
37   lr_type.iteration_plus_one      2738
0             lr_type.iteration      4032
1             lr_type.iteration      4032
15   lr_type.iteration_plus_one      2115
30   lr_type.iteration_plus_one      2199
8    lr_type.iteration_plus_one      1263
16   lr_type.iteration_plus_one      3062
42   lr_type.iteration_plus_one      2058
20   lr_type.iteration_plus_one      1934
17   lr_type.iteration_plus_one      2904
6    lr_type.iteration_plus_one       235
3           lr_type.sample_size      1362
7    lr_type.iteration_plus_one       531
33   lr_type.iteration_plus_one      2409
44   lr_type.iteration_plus_one      1946
43   lr_type.iteration_plus_one      3611
14   lr_type.iteration_plus_one      2469
26   lr_type.iteration_plus_one      2721
25   lr_type.iteration_plus_one      2146
34   lr_type.iteration_plus_one      1727
31   lr_type.iteration_plus_one      1994
32   lr_type.iteration_plus_one      1612
4       lr_type.ten_sample_size        74
5   lr_type.hundred_sample_size         2

                                              weights       epsilon  \
19  [[11.472057505397014435], [-4.4349784278962545…  1.000000e-05
46  [[11.152478136276841814], [-3.6847927278495271…  1.000000e-05
45  [[10.780086339913878238], [-3.7216570283533134…  1.000000e-05
38  [[10.780086339913878238], [-3.7216570283533134…  1.000000e-05
39  [[11.915738960490262956], [-4.1662803858045918…  1.000000e-05
24  [[11.571783751264503302], [-4.2273367710084355…  1.000000e-05
```

40	[[16.757890297389447553], [-6.5489032844729216…	1.000000e-05
11	[[9.087798687760632232], [-4.04964890730847365…	1.000000e-07
12	[[8.305635754643900189], [-4.08145059850403449…	1.000000e-08
41	[[12.931171030702525031], [-4.6407389412016888…	1.000000e-05
22	[[12.136361209371475922], [-4.3072039982668131…	1.000000e-05
28	[[8.850541831527898764], [-3.89118824863195484…	1.000000e-05
2	[[12.137409248966954996 4], [-4.520518798900213…	1.000000e-05
9	[[12.1374092489669549964], [-4.520518798900213…	1.000000e-05
13	[[7.781465198825748270 8], [-4.1149838488482304…	1.000000e-09
10	[[10.283574906574218732], [-4.1060086401810997…	1.000000e-06
23	[[10.550083355026051274], [-4.1522402634849284…	1.000000e-05
18	[[12.378667135942420577], [-5.2467260468350486…	1.000000e-05
27	[[11.456849077578187573], [-2.8345242831094986…	1.000000e-05
29	[[14.073646908470692887], [-5.5047098570100916…	1.000000e-05
36	[[11.727506805508466206], [-8.7890367450832210…	1.000000e-05
21	[[7.574352853182287649], [-4.53548774286644673…	1.000000e-05
35	[[7.998095415207553 54], [-3.90001097705347450 0…	1.000000e-05
37	[[11.810775725772269145], [-3.8200989106924167…	1.000000e-05
0	[[16.378906342486571375], [-5.5259574517590583…	1.000000e-05
1	[[16.378906342486571375], [-5.5259574517590583…	1.000000e-05
15	[[10.097329325595773509], [-0.4022719394966720…	1.000000e-05
30	[[12.710324024997548719], [-4.8963523723579962…	1.000000e-05
8	[[14.944116734687297383], [-5.9888301416859652…	1.000000e-04
16	[[12.0433165615157943], [-3.423846932597623296…	1.000000e-05
42	[[9.578820849453080831], [-4.04663715600940727…	1.000000e-05
20	[[13.077075012640899667], [-5.3331788140696238…	1.000000e-05
17	[[12.144219855790955124], [-5.0777700676349558…	1.000000e-05
6	[[20.268681216307229112], [-17.891141982398493…	1.000000e-02
3	[[3.239870586975073716], [-3.93183866315446415…	1.000000e-05
7	[[18.400300481674902137], [-9.9749477354584466…	1.000000e-03
33	[[6.573151944209476847], [-3.68075864668547587…	1.000000e-05
44	[[11.061917924354232654 5], [-4.980700263316878…	1.000000e-05
43	[[9.70288456997957947], [-4.517980142783009670…	1.000000e-05
14	[[1.216749001141672746 1], [-3.4149019369519555…	1.000000e-05
26	[[9.986283436440546812], [0.720063750748263452…	1.000000e-05
25	[[-2.6197091200274779232], [1.9603738862527356…	1.000000e-05
34	[[9.765787043289230331], [-5.42284373002951372…	1.000000e-05
31	[[15.0500309728559125205], [-5.361661631222361…	1.000000e-05
32	[[-1.0460067470590719306], [-4.565756503472974…	1.000000e-05
4	[[0.07851626733184168337], [-0.114624863637962…	1.000000e-05
5	[[0.00018731212820512820378], [-9.962399624765…	1.000000e-05

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
19	1.102	False	85.723087	0.742064	log
46	0.920	False	84.803401	0.741992	power*2
45	0.991	False	85.247168	0.741300	power*2
38	0.951	False	85.247168	0.741300	power*2

39	0.980	False	84.223758	0.741162	power*2
24	1.572	False	84.807459	0.740256	log
40	1.775	False	92.024182	0.740187	power*2
11	4.588	False	80.926339	0.739631	epsilon
12	7.677	False	80.216146	0.739354	epsilon
41	0.883	False	84.169551	0.739354	power*2
22	0.991	False	85.158078	0.738938	log
28	1.235	False	84.164349	0.738384	feature removal
2	1.110	False	86.235038	0.738105	learning rate
9	1.104	False	86.235038	0.738105	epsilon
13	15.933	False	79.885937	0.738036	epsilon
10	2.197	False	82.538558	0.737964	epsilon
23	0.867	False	81.190493	0.737131	log
18	1.327	False	83.420795	0.736923	log
27	1.427	False	86.799715	0.736504	feature removal
29	0.888	False	83.367648	0.736227	feature removal
36	1.425	False	85.761598	0.736157	power*2
21	1.140	False	82.468896	0.735949	log
35	1.016	False	84.943099	0.735532	power*2
37	1.107	False	85.890549	0.735323	power*2
0	5.061	False	97.099984	0.735186	all features
1	1.660	False	97.099984	0.735186	learning rate
15	0.878	False	83.854123	0.735117	log
30	0.855	False	85.039416	0.734769	feature removal
8	0.522	False	94.274794	0.734561	epsilon
16	1.237	False	87.890446	0.734283	log
42	0.833	False	84.638295	0.733376	power*2
20	0.812	False	85.514273	0.732755	log
17	1.434	False	85.180434	0.730737	log
6	0.151	False	134.658479	0.729835	epsilon
3	0.605	False	80.126535	0.729210	learning rate
7	0.365	False	109.385158	0.728721	epsilon
33	1.363	False	84.941279	0.728169	feature removal
44	0.919	False	85.532771	0.727539	power*2
43	1.697	False	91.446068	0.726500	power*2
14	1.117	False	86.710362	0.725452	log
26	1.080	False	86.886354	0.724344	feature removal
25	0.909	False	85.443068	0.722117	feature removal
34	0.682	False	84.911353	0.721982	feature removal
31	0.825	False	86.027563	0.720732	feature removal
32	0.645	False	86.142293	0.709404	feature removal
4	0.044	False	98.565776	0.536584	learning rate
5	0.001	False	99.809458	0.535056	learning rate

[83]:
```
for i in range(x_train_powerashmag.shape[1]):
    print (i,',column=>',ww_columns[i])
```

```
    x_train_power3 = power_n_feature(pd.DataFrame(x_train_powerashmag),i,3).
 ↪to_numpy()
    title = 'log Total phen/Proan-*2 Alkal of ash/mage-*3 {}'.
 ↪format(ww_columns[i])
    print(title)
    model_data = run_model('ww',title,learning_rate = 0, learning_rate_type =␣
 ↪learning_rate_type.iteration_plus_one ,max_iterations = 150000,
                           epsilon = 1e-5,x_train = x_train_power3 , y_train =␣
 ↪ww_y_train , model_data = model_data,variable = 'power*3')
show_sorted_model(model_data)
```

```
0 ,column=> Alcohol
log Total phen/Proan-*2 Alkal of ash/mage-*3 Alcohol
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
1 ,column=> Malic acid
log Total phen/Proan-*2 Alkal of ash/mage-*3 Malic acid
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
2 ,column=> Ash
log Total phen/Proan-*2 Alkal of ash/mage-*3 Ash
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
```

```
i======> 9
3 ,column=> Alkalinity of ash
log Total phen/Proan-*2 Alkal of ash/mage-*3 Alkalinity of ash
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
4 ,column=> Magnesium
log Total phen/Proan-*2 Alkal of ash/mage-*3 Magnesium
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
5 ,column=> Total phenols
log Total phen/Proan-*2 Alkal of ash/mage-*3 Total phenols
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
6 ,column=> Flavanoids
log Total phen/Proan-*2 Alkal of ash/mage-*3 Flavanoids
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
```

```
i======> 9
7 ,column=> Nonflavanoid phenols
log Total phen/Proan-*2 Alkal of ash/mage-*3 Nonflavanoid phenols
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
8 ,column=> Proanthocyanins
log Total phen/Proan-*2 Alkal of ash/mage-*3 Proanthocyanins
```

```
<ipython-input-59-bea85ad8211b>:100: RuntimeWarning: overflow encountered in exp
  return 1 / (1 + np.exp(float128(-arg)))
```

```
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
9 ,column=> Hue
log Total phen/Proan-*2 Alkal of ash/mage-*3 Hue
i======> 0
i======> 1
i======> 2
i======> 3
i======> 4
i======> 5
i======> 6
i======> 7
i======> 8
i======> 9
```

[83]:

```
   model_name                                        description  \
19         ww                                 log Total phenols
46         ww          log Total phen/Proan-*2 Alkal of ash/mage
45         ww          log Total phen/Proan-*2 Alkalinity of ash
38         ww          log Total phen-Proan-*2 Alkalinity of ash
39         ww                   log Total phen-Proan-*2 Magnesium
```

49	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 Ash
24	ww	log Total phenols-Proanthocyanins
40	ww	log Total phen-Proan-*2 Total phenols
52	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 T…
11	ww	whole model-epsilon:1e-07
12	ww	whole model-epsilon:1e-08
41	ww	log Total phen-Proan-*2 Flavanoids
22	ww	log Proanthocyanins
51	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 M…
28	ww	log Total phen-Proan-no Alkalinity of ash
50	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 A…
9	ww	whole model-epsilon:1e-05
2	ww	whole model-lr:lr_type.iteration_plus_one
13	ww	whole model-epsilon:1e-09
10	ww	whole model-epsilon:1e-06
23	ww	log Hue
18	ww	log Magnesium
27	ww	log Total phen-Proan-no Ash
29	ww	log Total phen-Proan-no Magnesium
36	ww	log Total phen-Proan-*2 Malic acid
21	ww	log Nonflavanoid phenols
48	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 M…
35	ww	log Total phen-Proan-*2 Alcohol
37	ww	log Total phen-Proan-*2 Ash
0	ww	whole model
1	ww	whole model-lr:lr_type.iteration
15	ww	log Malic acid
30	ww	log Total phen-Proan-no Total phenols
8	ww	whole model-epsilon:0.0001
47	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 A…
16	ww	log Ash
42	ww	log Total phen-Proan-*2 Nonflavanoid phenols
20	ww	log Flavanoids
55	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 P…
17	ww	log Alkalinity of ash
6	ww	whole model-epsilon:0.01
53	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 F…
3	ww	whole model-lr:lr_type.sample_size
7	ww	whole model-epsilon:0.001
33	ww	log Total phen-Proan-no Proanthocyanins
44	ww	log Total phen-Proan-*2 Hue
43	ww	log Total phen-Proan-*2 Proanthocyanins
14	ww	log Alcohol
54	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 N…
26	ww	log Total phen-Proan-no Malic acid
56	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 Hue
25	ww	log Total phen-Proan-no Alcohol

```
34         ww                       log Total phen-Proan-no Hue
31         ww                log Total phen-Proan-no Flavanoids
32         ww      log Total phen-Proan-no Nonflavanoid phenols
4          ww            whole model-lr:lr_type.ten_sample_size
5          ww        whole model-lr:lr_type.hundred_sample_size

                  learning_rate iteration  \
19   lr_type.iteration_plus_one      2743
46   lr_type.iteration_plus_one      2189
45   lr_type.iteration_plus_one      2318
38   lr_type.iteration_plus_one      2318
39   lr_type.iteration_plus_one      2387
49   lr_type.iteration_plus_one      2386
24   lr_type.iteration_plus_one      2523
40   lr_type.iteration_plus_one      3546
52   lr_type.iteration_plus_one      2928
11   lr_type.iteration_plus_one     10464
12   lr_type.iteration_plus_one     19610
41   lr_type.iteration_plus_one      2133
22   lr_type.iteration_plus_one      2439
51   lr_type.iteration_plus_one      2077
28   lr_type.iteration_plus_one      2267
50   lr_type.iteration_plus_one      2115
9    lr_type.iteration_plus_one      2753
2    lr_type.iteration_plus_one      2753
13   lr_type.iteration_plus_one     36990
10   lr_type.iteration_plus_one      5496
23   lr_type.iteration_plus_one      2118
18   lr_type.iteration_plus_one      2205
27   lr_type.iteration_plus_one      2961
29   lr_type.iteration_plus_one      2268
36   lr_type.iteration_plus_one      2722
21   lr_type.iteration_plus_one      1844
48   lr_type.iteration_plus_one      2316
35   lr_type.iteration_plus_one      2334
37   lr_type.iteration_plus_one      2738
0             lr_type.iteration      4032
1             lr_type.iteration      4032
15   lr_type.iteration_plus_one      2115
30   lr_type.iteration_plus_one      2199
8    lr_type.iteration_plus_one      1263
47   lr_type.iteration_plus_one      1912
16   lr_type.iteration_plus_one      3062
42   lr_type.iteration_plus_one      2058
20   lr_type.iteration_plus_one      1934
55   lr_type.iteration_plus_one     10504
17   lr_type.iteration_plus_one      2904
```

```
6    lr_type.iteration_plus_one       235
53   lr_type.iteration_plus_one      1885
3           lr_type.sample_size      1362
7    lr_type.iteration_plus_one       531
33   lr_type.iteration_plus_one      2409
44   lr_type.iteration_plus_one      1946
43   lr_type.iteration_plus_one      3611
14   lr_type.iteration_plus_one      2469
54   lr_type.iteration_plus_one      1468
26   lr_type.iteration_plus_one      2721
56   lr_type.iteration_plus_one      1617
25   lr_type.iteration_plus_one      2146
34   lr_type.iteration_plus_one      1727
31   lr_type.iteration_plus_one      1994
32   lr_type.iteration_plus_one      1612
4       lr_type.ten_sample_size        74
5   lr_type.hundred_sample_size         2

                                              weights       epsilon  \
19  [[11.472057505397014435], [-4.4349784278962545…  1.000000e-05
46  [[11.152478136276841814], [-3.6847927278495271…  1.000000e-05
45  [[10.780086339913878238], [-3.7216570283533134…  1.000000e-05
38  [[10.780086339913878238], [-3.7216570283533134…  1.000000e-05
39  [[11.915738960490262956], [-4.1662803858045918…  1.000000e-05
49  [[10.925562554548044123], [-2.8587902436059984…  1.000000e-05
24  [[11.571783751264503302], [-4.2273367710084355…  1.000000e-05
40  [[16.757890297389447553], [-6.5489032844729216…  1.000000e-05
52  [[16.040815800184150163], [-5.4694767252486631…  1.000000e-05
11  [[9.087798687760632232], [-4.04964890730847365…  1.000000e-07
12  [[8.305635754643900189], [-4.08145059850403449…  1.000000e-08
41  [[12.931171030702525031], [-4.6407389412016888…  1.000000e-05
22  [[12.136361209371475922], [-4.3072039982668131…  1.000000e-05
51  [[12.869863294781712618], [-4.6242760645604920…  1.000000e-05
28  [[8.850541831527898764], [-3.89118824863195484…  1.000000e-05
50  [[9.768092957788114001], [-3.70582777449206642…  1.000000e-05
9   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
2   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
13  [[7.7814651988257482708], [-4.1149838488482304…  1.000000e-09
10  [[10.283574906574218732], [-4.1060086401810997…  1.000000e-06
23  [[10.550083355026051274], [-4.1522402634849284…  1.000000e-05
18  [[12.378667135942420577], [-5.2467260468350486…  1.000000e-05
27  [[11.456849077578187573], [-2.8345242831094986…  1.000000e-05
29  [[14.073646908470692887], [-5.5047098570100916…  1.000000e-05
36  [[11.727506805508466206], [-8.7890367450832210…  1.000000e-05
21  [[7.574352853182287649], [-4.53548774286644673…  1.000000e-05
48  [[10.902156765272937098], [-13.566703259924261…  1.000000e-05
35  [[7.99809541520755354], [-3.900010977053474500…  1.000000e-05
```

37	[[11.810775725772269145], [-3.8200989106924167…	1.000000e-05
0	[[16.378906342486571375], [-5.5259574517590583…	1.000000e-05
1	[[16.378906342486571375], [-5.5259574517590583…	1.000000e-05
15	[[10.097329325595773509], [-0.4022719394966720…	1.000000e-05
30	[[12.710324024997548719], [-4.8963523723579962…	1.000000e-05
8	[[14.944116734687297383], [-5.9888301416859652…	1.000000e-04
47	[[6.187533458939720453], [-3.09896354987425925…	1.000000e-05
16	[[12.0433165615157943], [-3.423846932597623296…	1.000000e-05
42	[[9.578820849453080831], [-4.04663715600940727…	1.000000e-05
20	[[13.077075012640899667], [-5.3331788140696238…	1.000000e-05
55	[[5.7804716020247181563], [-2.8445541899269348…	1.000000e-05
17	[[12.144219855790955124], [-5.0777700676349558…	1.000000e-05
6	[[20.268681216307229112], [-17.891141982398493…	1.000000e-02
53	[[13.954016667380731965], [-4.6352743287717679…	1.000000e-05
3	[[3.239870586975073716], [-3.93183866315446415…	1.000000e-05
7	[[18.400300481674902137], [-9.9749477354584466…	1.000000e-03
33	[[6.573151944209476847], [-3.68075864668547587…	1.000000e-05
44	[[11.0619179243542326545], [-4.980700263316878…	1.000000e-05
43	[[9.70288456997957947], [-4.517980142783009670…	1.000000e-05
14	[[1.2167490011416727461], [-3.4149019369519555…	1.000000e-05
54	[[6.0210858160443539245], [-3.8782577792302064…	1.000000e-05
26	[[9.986283436440546812], [0.720063750748263452…	1.000000e-05
56	[[9.907532219817162206], [-4.93994452301035420…	1.000000e-05
25	[[-2.6197091200274779232], [1.9603738862527356…	1.000000e-05
34	[[9.765787043289230331], [-5.42284373002951372…	1.000000e-05
31	[[15.0500309728559125205], [-5.361661631222361…	1.000000e-05
32	[[-1.0460067470590719306], [-4.565756503472974…	1.000000e-05
4	[[0.07851626733184168337], [-0.114624863637962…	1.000000e-05
5	[[0.00018731212820512820378], [-9.962399624765…	1.000000e-05

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
19	1.102	False	85.723087	0.742064	log
46	0.920	False	84.803401	0.741992	power*2
45	0.991	False	85.247168	0.741300	power*2
38	0.951	False	85.247168	0.741300	power*2
39	0.980	False	84.223758	0.741162	power*2
49	0.966	False	85.998220	0.740397	power*3
24	1.572	False	84.807459	0.740256	log
40	1.775	False	92.024182	0.740187	power*2
52	1.208	False	89.686677	0.739634	power*3
11	4.588	False	80.926339	0.739631	epsilon
12	7.677	False	80.216146	0.739354	epsilon
41	0.883	False	84.169551	0.739354	power*2
22	0.991	False	85.158078	0.738938	log
51	0.851	False	83.504929	0.738726	power*3
28	1.235	False	84.164349	0.738384	feature removal
50	0.929	False	83.642757	0.738171	power*3

9	1.104	False	86.235038	0.738105	epsilon
2	1.110	False	86.235038	0.738105	learning rate
13	15.933	False	79.885937	0.738036	epsilon
10	2.197	False	82.538558	0.737964	epsilon
23	0.867	False	81.190493	0.737131	log
18	1.327	False	83.420795	0.736923	log
27	1.427	False	86.799715	0.736504	feature removal
29	0.888	False	83.367648	0.736227	feature removal
36	1.425	False	85.761598	0.736157	power*2
21	1.140	False	82.468896	0.735949	log
48	0.945	False	85.927533	0.735740	power*3
35	1.016	False	84.943099	0.735532	power*2
37	1.107	False	85.890549	0.735323	power*2
0	5.061	False	97.099984	0.735186	all features
1	1.660	False	97.099984	0.735186	learning rate
15	0.878	False	83.854123	0.735117	log
30	0.855	False	85.039416	0.734769	feature removal
8	0.522	False	94.274794	0.734561	epsilon
47	0.916	False	85.599770	0.734283	power*3
16	1.237	False	87.890446	0.734283	log
42	0.833	False	84.638295	0.733376	power*2
20	0.812	False	85.514273	0.732755	log
55	4.129	False	84.697588	0.732476	power*3
17	1.434	False	85.180434	0.730737	log
6	0.151	False	134.658479	0.729835	epsilon
53	0.785	False	86.114857	0.729417	power*3
3	0.605	False	80.126535	0.729210	learning rate
7	0.365	False	109.385158	0.728721	epsilon
33	1.363	False	84.941279	0.728169	feature removal
44	0.919	False	85.532771	0.727539	power*2
43	1.697	False	91.446068	0.726500	power*2
14	1.117	False	86.710362	0.725452	log
54	0.631	False	85.404313	0.725387	power*3
26	1.080	False	86.886354	0.724344	feature removal
56	0.670	False	85.844925	0.722816	power*3
25	0.909	False	85.443068	0.722117	feature removal
34	0.682	False	84.911353	0.721982	feature removal
31	0.825	False	86.027563	0.720732	feature removal
32	0.645	False	86.142293	0.709404	feature removal
4	0.044	False	98.565776	0.536584	learning rate
5	0.001	False	99.809458	0.535056	learning rate

[84]:
```
show_sorted_model(model_data)
```

[84]:

	model_name	description \
19	ww	log Total phenols
46	ww	log Total phen/Proan-*2 Alkal of ash/mage

45	ww	log Total phen/Proan-*2 Alkalinity of ash
38	ww	log Total phen-Proan-*2 Alkalinity of ash
39	ww	log Total phen-Proan-*2 Magnesium
49	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 Ash
24	ww	log Total phenols-Proanthocyanins
40	ww	log Total phen-Proan-*2 Total phenols
52	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 T…
11	ww	whole model-epsilon:1e-07
12	ww	whole model-epsilon:1e-08
41	ww	log Total phen-Proan-*2 Flavanoids
22	ww	log Proanthocyanins
51	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 M…
28	ww	log Total phen-Proan-no Alkalinity of ash
50	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 A…
9	ww	whole model-epsilon:1e-05
2	ww	whole model-lr:lr_type.iteration_plus_one
13	ww	whole model-epsilon:1e-09
10	ww	whole model-epsilon:1e-06
23	ww	log Hue
18	ww	log Magnesium
27	ww	log Total phen-Proan-no Ash
29	ww	log Total phen-Proan-no Magnesium
36	ww	log Total phen-Proan-*2 Malic acid
21	ww	log Nonflavanoid phenols
48	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 M…
35	ww	log Total phen-Proan-*2 Alcohol
37	ww	log Total phen-Proan-*2 Ash
0	ww	whole model
1	ww	whole model-lr:lr_type.iteration
15	ww	log Malic acid
30	ww	log Total phen-Proan-no Total phenols
8	ww	whole model-epsilon:0.0001
47	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 A…
16	ww	log Ash
42	ww	log Total phen-Proan-*2 Nonflavanoid phenols
20	ww	log Flavanoids
55	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 P…
17	ww	log Alkalinity of ash
6	ww	whole model-epsilon:0.01
53	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 F…
3	ww	whole model-lr:lr_type.sample_size
7	ww	whole model-epsilon:0.001
33	ww	log Total phen-Proan-no Proanthocyanins
44	ww	log Total phen-Proan-*2 Hue
43	ww	log Total phen-Proan-*2 Proanthocyanins
14	ww	log Alcohol
54	ww	log Total phen/Proan-*2 Alkal of ash/mage-*3 N…

```
26         ww                 log Total phen-Proan-no Malic acid
56         ww   log Total phen/Proan-*2 Alkal of ash/mage-*3 Hue
25         ww                    log Total phen-Proan-no Alcohol
34         ww                        log Total phen-Proan-no Hue
31         ww                 log Total phen-Proan-no Flavanoids
32         ww       log Total phen-Proan-no Nonflavanoid phenols
4          ww             whole model-lr:lr_type.ten_sample_size
5          ww         whole model-lr:lr_type.hundred_sample_size

                  learning_rate iteration  \
19   lr_type.iteration_plus_one      2743
46   lr_type.iteration_plus_one      2189
45   lr_type.iteration_plus_one      2318
38   lr_type.iteration_plus_one      2318
39   lr_type.iteration_plus_one      2387
49   lr_type.iteration_plus_one      2386
24   lr_type.iteration_plus_one      2523
40   lr_type.iteration_plus_one      3546
52   lr_type.iteration_plus_one      2928
11   lr_type.iteration_plus_one     10464
12   lr_type.iteration_plus_one     19610
41   lr_type.iteration_plus_one      2133
22   lr_type.iteration_plus_one      2439
51   lr_type.iteration_plus_one      2077
28   lr_type.iteration_plus_one      2267
50   lr_type.iteration_plus_one      2115
9    lr_type.iteration_plus_one      2753
2    lr_type.iteration_plus_one      2753
13   lr_type.iteration_plus_one     36990
10   lr_type.iteration_plus_one      5496
23   lr_type.iteration_plus_one      2118
18   lr_type.iteration_plus_one      2205
27   lr_type.iteration_plus_one      2961
29   lr_type.iteration_plus_one      2268
36   lr_type.iteration_plus_one      2722
21   lr_type.iteration_plus_one      1844
48   lr_type.iteration_plus_one      2316
35   lr_type.iteration_plus_one      2334
37   lr_type.iteration_plus_one      2738
0             lr_type.iteration      4032
1             lr_type.iteration      4032
15   lr_type.iteration_plus_one      2115
30   lr_type.iteration_plus_one      2199
8    lr_type.iteration_plus_one      1263
47   lr_type.iteration_plus_one      1912
16   lr_type.iteration_plus_one      3062
42   lr_type.iteration_plus_one      2058
```

```
20   lr_type.iteration_plus_one      1934
55   lr_type.iteration_plus_one     10504
17   lr_type.iteration_plus_one      2904
6    lr_type.iteration_plus_one       235
53   lr_type.iteration_plus_one      1885
3           lr_type.sample_size      1362
7    lr_type.iteration_plus_one       531
33   lr_type.iteration_plus_one      2409
44   lr_type.iteration_plus_one      1946
43   lr_type.iteration_plus_one      3611
14   lr_type.iteration_plus_one      2469
54   lr_type.iteration_plus_one      1468
26   lr_type.iteration_plus_one      2721
56   lr_type.iteration_plus_one      1617
25   lr_type.iteration_plus_one      2146
34   lr_type.iteration_plus_one      1727
31   lr_type.iteration_plus_one      1994
32   lr_type.iteration_plus_one      1612
4       lr_type.ten_sample_size        74
5   lr_type.hundred_sample_size         2

                                                weights       epsilon  \
19  [[11.472057505397014435], [-4.4349784278962545…  1.000000e-05
46  [[11.152478136276841814], [-3.6847927278495271…  1.000000e-05
45  [[10.780086339913878238], [-3.7216570283533134…  1.000000e-05
38  [[10.780086339913878238], [-3.7216570283533134…  1.000000e-05
39  [[11.915738960490262956], [-4.1662803858045918…  1.000000e-05
49  [[10.925562554548044123], [-2.8587902436059984…  1.000000e-05
24  [[11.571783751264503302], [-4.2273367710084355…  1.000000e-05
40  [[16.757890297389447553], [-6.5489032844729216…  1.000000e-05
52  [[16.040815800184150163], [-5.4694767252486631…  1.000000e-05
11  [[9.087798687760632232], [-4.04964890730847365…  1.000000e-07
12  [[8.305635754643900189], [-4.08145059850403449…  1.000000e-08
41  [[12.931171030702525031], [-4.6407389412016888…  1.000000e-05
22  [[12.136361209371475922], [-4.3072039982668131…  1.000000e-05
51  [[12.869863294781712618], [-4.6242760645604920…  1.000000e-05
28  [[8.850541831527898764], [-3.89118824863195484…  1.000000e-05
50  [[9.768092957788114001], [-3.70582777449206642…  1.000000e-05
9   [[12.137409248966954996 4], [-4.520518798900213…  1.000000e-05
2   [[12.1374092489669549964], [-4.520518798900213…  1.000000e-05
13  [[7.781465198825748270 8], [-4.1149838488482304…  1.000000e-09
10  [[10.283574906574218732], [-4.1060086401810997…  1.000000e-06
23  [[10.550083355026051274], [-4.1522402634849284…  1.000000e-05
18  [[12.378667135942420577], [-5.2467260468350486…  1.000000e-05
27  [[11.456849077578187573], [-2.8345242831094986…  1.000000e-05
29  [[14.073646908470692887], [-5.5047098570100916…  1.000000e-05
36  [[11.727506805508466206], [-8.7890367450832210…  1.000000e-05
```

21	[[7.574352853182287649], [-4.53548774286644673…	1.000000e-05
48	[[10.902156765272937098], [-13.566703259924261…	1.000000e-05
35	[[7.99809541520755354], [-3.900010977053474500…	1.000000e-05
37	[[11.810775725772269145], [-3.8200989106924167…	1.000000e-05
0	[[16.378906342486571375], [-5.5259574517590583…	1.000000e-05
1	[[16.378906342486571375], [-5.5259574517590583…	1.000000e-05
15	[[10.097329325595773509], [-0.4022719394966720…	1.000000e-05
30	[[12.710324024997548719], [-4.8963523723579962…	1.000000e-05
8	[[14.944116734687297383], [-5.9888301416859652…	1.000000e-04
47	[[6.187533458939720453], [-3.09896354987425925…	1.000000e-05
16	[[12.0433165615157943], [-3.423846932597623296…	1.000000e-05
42	[[9.578820849453080831], [-4.04663715600940727…	1.000000e-05
20	[[13.077075012640899667], [-5.3331788140696238…	1.000000e-05
55	[[5.7804716020247181563], [-2.8445541899269348…	1.000000e-05
17	[[12.144219855790955124], [-5.0777700676349558…	1.000000e-05
6	[[20.268681216307229112], [-17.891141982398493…	1.000000e-02
53	[[13.954016667380731965], [-4.6352743287717679…	1.000000e-05
3	[[3.239870586975073716], [-3.93183866315446415…	1.000000e-05
7	[[18.400300481674902137], [-9.9749477354584466…	1.000000e-03
33	[[6.573151944209476847], [-3.68075864668547587…	1.000000e-05
44	[[11.0619179243542326545], [-4.980700263316878…	1.000000e-05
43	[[9.70288456997957947], [-4.517980142783009670…	1.000000e-05
14	[[1.2167490011416727461], [-3.4149019369519555…	1.000000e-05
54	[[6.0210858160443539245], [-3.8782577792302064…	1.000000e-05
26	[[9.986283436440546812], [0.720063750748263452…	1.000000e-05
56	[[9.907532219817162206], [-4.93994452301035420…	1.000000e-05
25	[[-2.6197091200274779232], [1.9603738862527356…	1.000000e-05
34	[[9.765787043289230331], [-5.42284373002951372…	1.000000e-05
31	[[15.0500309728559125205], [-5.361661631222361…	1.000000e-05
32	[[-1.0460067470590719306], [-4.565756503472974…	1.000000e-05
4	[[0.07851626733184168337], [-0.114624863637962…	1.000000e-05
5	[[0.00018731212820512820378], [-9.962399624765…	1.000000e-05

	elapsed_time	is_max_reached	loss	accuracy_kfold	variable
19	1.102	False	85.723087	0.742064	log
46	0.920	False	84.803401	0.741992	power*2
45	0.991	False	85.247168	0.741300	power*2
38	0.951	False	85.247168	0.741300	power*2
39	0.980	False	84.223758	0.741162	power*2
49	0.966	False	85.998220	0.740397	power*3
24	1.572	False	84.807459	0.740256	log
40	1.775	False	92.024182	0.740187	power*2
52	1.208	False	89.686677	0.739634	power*3
11	4.588	False	80.926339	0.739631	epsilon
12	7.677	False	80.216146	0.739354	epsilon
41	0.883	False	84.169551	0.739354	power*2
22	0.991	False	85.158078	0.738938	log

51	0.851	False	83.504929	0.738726	power*3
28	1.235	False	84.164349	0.738384	feature removal
50	0.929	False	83.642757	0.738171	power*3
9	1.104	False	86.235038	0.738105	epsilon
2	1.110	False	86.235038	0.738105	learning rate
13	15.933	False	79.885937	0.738036	epsilon
10	2.197	False	82.538558	0.737964	epsilon
23	0.867	False	81.190493	0.737131	log
18	1.327	False	83.420795	0.736923	log
27	1.427	False	86.799715	0.736504	feature removal
29	0.888	False	83.367648	0.736227	feature removal
36	1.425	False	85.761598	0.736157	power*2
21	1.140	False	82.468896	0.735949	log
48	0.945	False	85.927533	0.735740	power*3
35	1.016	False	84.943099	0.735532	power*2
37	1.107	False	85.890549	0.735323	power*2
0	5.061	False	97.099984	0.735186	all features
1	1.660	False	97.099984	0.735186	learning rate
15	0.878	False	83.854123	0.735117	log
30	0.855	False	85.039416	0.734769	feature removal
8	0.522	False	94.274794	0.734561	epsilon
47	0.916	False	85.599770	0.734283	power*3
16	1.237	False	87.890446	0.734283	log
42	0.833	False	84.638295	0.733376	power*2
20	0.812	False	85.514273	0.732755	log
55	4.129	False	84.697588	0.732476	power*3
17	1.434	False	85.180434	0.730737	log
6	0.151	False	134.658479	0.729835	epsilon
53	0.785	False	86.114857	0.729417	power*3
3	0.605	False	80.126535	0.729210	learning rate
7	0.365	False	109.385158	0.728721	epsilon
33	1.363	False	84.941279	0.728169	feature removal
44	0.919	False	85.532771	0.727539	power*2
43	1.697	False	91.446068	0.726500	power*2
14	1.117	False	86.710362	0.725452	log
54	0.631	False	85.404313	0.725387	power*3
26	1.080	False	86.886354	0.724344	feature removal
56	0.670	False	85.844925	0.722816	power*3
25	0.909	False	85.443068	0.722117	feature removal
34	0.682	False	84.911353	0.721982	feature removal
31	0.825	False	86.027563	0.720732	feature removal
32	0.645	False	86.142293	0.709404	feature removal
4	0.044	False	98.565776	0.536584	learning rate
5	0.001	False	99.809458	0.535056	learning rate

[85]:
```
(model_data.sort_values(by=['accuracy_kfold'], ascending=False)).
↪to_csv('white_whine_models.csv', index=False)
```

[86]:
```python
import os
cwd = os.getcwd()
print(cwd)
```

```
/content
```

[87]:
```python
model_data.loc[model_data['variable'] == 'epsilon']
```

[87]:
```
   model_name                  description                learning_rate  \
6          ww    whole model-epsilon:0.01  lr_type.iteration_plus_one
7          ww   whole model-epsilon:0.001  lr_type.iteration_plus_one
8          ww  whole model-epsilon:0.0001  lr_type.iteration_plus_one
9          ww   whole model-epsilon:1e-05  lr_type.iteration_plus_one
10         ww   whole model-epsilon:1e-06  lr_type.iteration_plus_one
11         ww   whole model-epsilon:1e-07  lr_type.iteration_plus_one
12         ww   whole model-epsilon:1e-08  lr_type.iteration_plus_one
13         ww   whole model-epsilon:1e-09  lr_type.iteration_plus_one

   iteration                                            weights       epsilon  \
6        235  [[20.268681216307229112], [-17.891141982398493…  1.000000e-02
7        531  [[18.400300481674902137], [-9.9749477354584466…  1.000000e-03
8       1263  [[14.944116734687297383], [-5.9888301416859652…  1.000000e-04
9       2753  [[12.137409248966954996 4], [-4.520518798900213…  1.000000e-05
10      5496  [[10.283574906574218732], [-4.1060086401810997…  1.000000e-06
11     10464  [[9.087798687760632232], [-4.0496489073084736 5…  1.000000e-07
12     19610  [[8.305635754643900189], [-4.0814505985040344 9…  1.000000e-08
13     36990  [[7.781465198825748270 8], [-4.1149838488482304…  1.000000e-09

    elapsed_time is_max_reached        loss  accuracy_kfold variable
6          0.151          False  134.658479        0.729835  epsilon
7          0.365          False  109.385158        0.728721  epsilon
8          0.522          False   94.274794        0.734561  epsilon
9          1.104          False   86.235038        0.738105  epsilon
10         2.197          False   82.538558        0.737964  epsilon
11         4.588          False   80.926339        0.739631  epsilon
12         7.677          False   80.216146        0.739354  epsilon
13        15.933          False   79.885937        0.738036  epsilon
```