سپهر آزردار 810199357

MI

فاز2:

پیش پردازش:

۱. دو روش برای حل مشکل Values Missing ،حذف کل ستون و پر کردن مقادیر خالی با آمارهها (برای مثال مد) میباشد. باقی روشها را توضیح دهید و مقایسه کنید.

- Deleting Rows with missing values
  - Des:
    - deleting the rows or columns having null values. If columns have more than half of the rows as null then the entire column can be dropped. The rows which are having one or more columns values as null can also be dropped.
  - Pros:
    - Create a robust model.
  - Cons:
    - Loss a lot of info.
    - Works poorly if a big portion of values is missing in comparison to the complete df
- Impute missing values for continuous variable
  - Des:
    - replaced with the mean, median, or mode of remaining values in the column.
  - Pros:
    - This method can prevent the loss of data compared to the earlier method.
    - Works well with a small dataset and is easy to implement
  - Cons:
    - Works only with numerical continuous variables.
    - Can cause data leakage
    - Do not factor the covariance between features.
- Impute missing values for categorical variable
  - Des:
    - the missing values can be replaced with the most frequent category. If the number of missing values is very large then it can be replaced with a new category.
  - Pros:
    - This method can prevent the loss of data compared to the earlier method.
    - Works well with a small dataset and is easy to implement.
  - Cons
    - Works only with categorical variables.
    - Addition of new features may result in poor performance

- Other Imputation Methods
    - For the data variable having longitudinal behavior.
      The last valid observation to fill the missing value.
      Last observation carried forward (LOCF) method.

    - For the time-series dataset variable, it makes sense to
      use the interpolation of the variable before and after a timestamp for a missing value
- Using Algorithms that support missing values
    - The k-NN, ignore a column from a distance measure when a value is missing
    - Naive Bayes
    - The sklearn implementations of naive Bayes and k-Nearest Neighbors and RF in Python do not support the presence of the missing values.
    - RandomForest that works well on non-linear and categorical data. It adapts to the data structure taking into consideration the high variance or the bias, producing better results on large datasets.
- Prediction of missing values
    - use the correlation advantage of the variable containing the missing value and other variables. Using the other features which don't have nulls can be used to predict missing values.
    - The regression or classification model can be used for the prediction of missing values depending on the nature (categorical or continuous) of the feature having missing value.
- Imputation using Deep Learning Library — Datawig
    - This method works very well with categorical, continuous, and non-numerical features
      Use deep NN.
    - Pros:
        - Quite accurate compared to other methods.
        - It supports CPUs and GPUs.
    - Cons:
        - Can be quite slow with large datasets.

- Replacing with Previous Value – Forward Fill
- Replacing with Next Value – Backward Fil

۲. بر اساس نتایج فاز قبل، کدام دادهها بیشترین میزان داده گم شده را دارند؟ برای تمامی ویژگیها مشکل داده های گم شده را با کمک روشهای مطرح شده حل کنید.

۳. در ویژگیهای عددی، normalizing یا standardizing به چه منظور انجام میشود؟ در این پروژه نیاز به انجام این کار هست؟

به بقیه روش ها به جز tree & forest کمک میکند.

(1)No, scaling is not necessary for random forests, (2) Random Forest is a tree-based model and hence does not require feature scaling.

At what modelling step do we apply feature scaling?

It is important to mention that before applying any sort of data normalisation, we first need to split our initial dataset into training and testing sets. Don't forget that testing data points represent real-world data. perform normalisation on testing instances as well, but this time using the mean and standard deviation of training explanatory variables

Algorithms that compute the distance between the features are biased towards numerically larger values if the data is not scaled.

having features of different scale and range of values in the dataset lower our performance.

Tree-based algorithms are fairly insensitive to the scale of the features

DES:

The main goal of normalization is to make the data homogenous over all records and fields

Whereas data standardization is the process of placing dissimilar features on the same scale.

rescaling the attributes in such a way that their mean is 0 and standard deviation becomes 1.

WHEN TO NORM:

X_new = (X - X_min)/(X_max - X_min)

Normalization of data is a type of Feature scaling and is only required when the data distribution is unknown or the data doesn't have Gaussian Distribution.

used when the data has a diversified scope and the algorithms on which the data are being trained do not make presumptions about the data distribution such as Artificial Neural Network.
When To Standardize Data?

X_new = (X - mean)/Std

used for multivariate analysis i.e. when we want all the variables of comparable units
more effective when applied to Gaussian distribution. This technique comes in handy when the data has varying ratios and the algorithms used, make assumptions about the data distribution like Logistic Regression, Linear Discriminant Analysis,

NORM VS STD. :
Normalization scales in a range of [0,1] or [-1,1]. Standardization is not bounded by range.
Normalization is highly affected by outliers. Standardization is slightly affected by outliers.

۴. برای استفاده ویژگیهای دسته ای، که معمولا بصورت یک string یا object در مجموعه داده ذخیره شدهاند، در آموزش مدل چه پیش پردازشهایی کارگر هستند؟ آیا همه دادههای دسته ای نیازمند این روشها هستند؟

Ordinal categorical variables:

Using map() function
Using Label Encoder

Nominal categorical variables:

One-Hot Encoding; One-Hot Encoding; drop_first=True in order to avoid the problem of Multicollinearity.

۵. آیا امکان حذف برخی ستونها وجود دارد؟ چرا؟

خیر، برای اینکه (مخصوصا در داده های کوچک) loss of data خواهیم داشت. در صورتی که بخش عظیمی از ستون داده نباشد. تاثیرات منفی این روش کمتر است.

۶. برای آموزش و در نهایت ارزیابی مدل یادگیری ماشین نیاز است که داده ها را به دو دسته test و train تقسیم کنیم. نسبت این تقسیم به چه صورت است؟ چه روشهای برای تقسیم و ساخت این دو دسته وجود دارد؟

DES:

The simplest way to split the modelling dataset into training and testing sets is to assign 2/3 data points to the former and the remaining one-third to the latter.; a 80:20 ratio for training:testing sets.

certain situations you should consider creating an extra set called the validation set.is usually required when apart from model performance we also need to choose among many models

HOW TO:

use pandas DataFrames' method sample (): Return a random sample of items from an axis of object.

Most common: Sklearn's method called train_test_split () : Split arrays or matrices into random train and test subsets: training_data, testing_data = train_test_split(df, test_size=0.2, random_state=25)

NumPy 's method Rand (): Return a sample (or samples) from the "standard normal" distribution.
We first create mask which is a numpy array that contains boolean values that were computed by comparing a random float numbers in the range between 0 and 1 with the fraction we want to keep for the training set. however that this approach will approximately give a 80:20 ration

pre-processing steps such as scaling or normalisation. You must be careful when doing so, since you need to avoid introducing future information into your training set.

۷ .گاهی علاوه بر دو دسته بالا یک دسته سومی هم وجود دارد. در مورد این دسته (validation) توضیح دهید.

در واقع از این نوع داده برای hyper tunning استفاده میشود.

certain situations you should consider creating an extra set called the validation set.is usually required when apart from model performance we also need to choose among many models

فاز سوم:

آموزش و ارزیابی:

۱. دقت هر مدل را بر اساس matrix confusion رسم شده بدستآورید و نتایج را توضیح دهید.

۲. برای مدلهایی که پارامترهای زیادی دارند با کمک تابع GridSearchCV ،مقادیر بهینه برای پارامترها را بدست آورید.

Standardization isn't required for logistic regression. The main goal of standardizing features is to help convergence of the technique used for optimization

۳. در مورد underfitting و overfitting تحقیق کنید. آیا در مدلهای شما این پدیدهها رخ دادند؟

Tree_decision مدل ها ذاتا overfitting هستند.

This phenomenon occurs when a model performs really well on the data that we used to train it but it fails to generalise well to new, unseen data points. There are numerous reasons why this can happen — it could be due to the noise in data or it could be that the model learned to predict specific inputs rather than the predictive parameters that could help it make correct predictions. Typically, the higher the complexity of a model the higher the chance that it will be overfitted.

underfitting occurs when the model has poor performance even on the data that was used to train it. Usually, this means that the model is less complex than required in order to learn those parameters that can be proven to be predictive.

یادگیری جمعی:

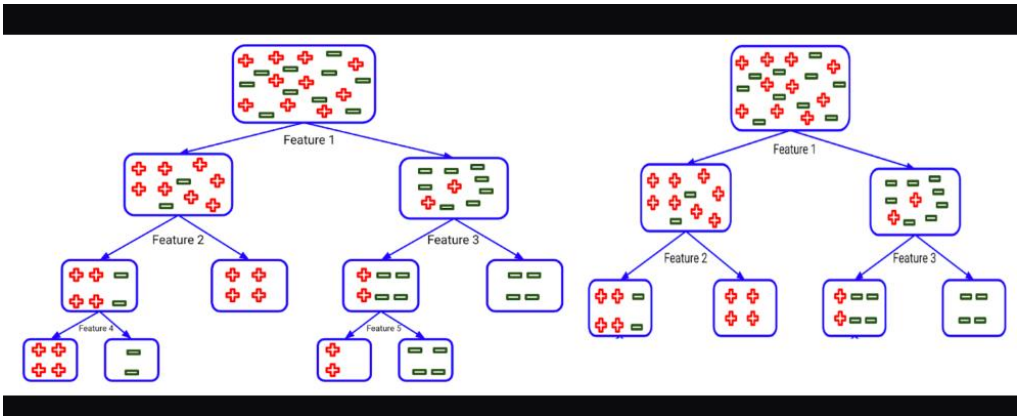
۱. در مورد حداقل دو عدد از هایپرپارامترهای این مدل مطالعه کنید و تاثیر تغییر این هایپرپارامترها را روی نتایجتان را با رسم نمودار و ذکر دقیق نتایج بسنجید.

- max_depth

  The max_depth of a tree in Random Forest is defined as the longest path between the root node and the leaf node; Using the max_depth parameter, I can limit up to what depth I want every tree in my random forest to grow.

  As the value grows, the model starts to overfit.

- min_sample_split



tells the decision tree in a random forest the minimum required number of observations in any given node in order to split it.

The is 2. This means that if any terminal node has more than two observations and is not a pure node, we can split it further into sub nodes. ***By increasing the value of the min_sample_split, we can reduce the number of splits that happen in the decision tree and therefore prevent the model from overfitting.*** As the value grows too much, the model starts to under fit.

- max_leaf_nodes

  If after splitting we have more terminal nodes than the specified number of terminal nodes, it will stop the splitting and the tree will not grow further.

- min_samples_leaf

  the minimum number of samples that should be present in the leaf node after splitting a node.

- n_estimators

  We might say that more trees should be able to produce a more generalized result; But by choosing more number of trees, the time complexity of the Random Forest model also increases. By increasing the value, the performance

of the model sharply increases and then stagnates at a certain level. Thus after it is just computational overhead. the performance of the model rises sharply and then saturates fairly quickly. the model performance reaches its max when the data provided is less than 0.2 fraction of the original dataset.

- max_sample (bootstrap sample)

  The max_samples hyperparameter determines what fraction of the original dataset is given to any individual tree;

- max_features

  number of maximum features provided to each tree in a random forest.

۲. نتایج این مدل را با مدل Tree Decision مقایسه کنید. در مورد bias و variance و ارتباط بین آن ها مطالعه کنید. به نظر شما از نظر هر کدام از دو مورد bias و variance یک مدل تنها Tree Decision بهتر عمل میکند یا یک مدل تجمیعی Forest Random؟ آیا نتایجی که به دستآوردید با نظرتان مطابقت دارد

prediction errors (bias and variance); There is a tradeoff between a model's ability to minimize bias and variance;

Bias:

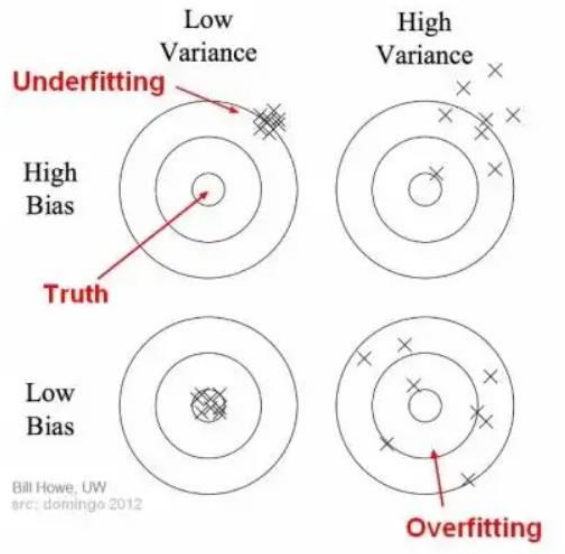
Bias is the difference between the average prediction of our model and the correct value which we are trying to predict. Model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to high error on training and test data.

Variance:

Variance is the variability of model prediction for a given data point or a value which tells us spread of our data; Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn't seen before. As a result, such models perform very well on training data but has high error rates on test data.

$$Err(x) = \left(E[\hat{f}(x)] - f(x)\right)^2 + E\left[\left(\hat{f}(x) - E[\hat{f}(x)]\right)^2\right] + \sigma_e^2$$

$$Err(x) = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$



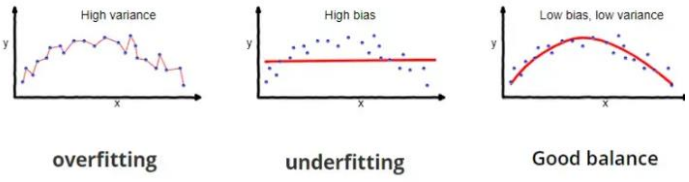
Irreducible error is the error that can't be reduced by creating good models. It is a measure of the amount of noise in our data.

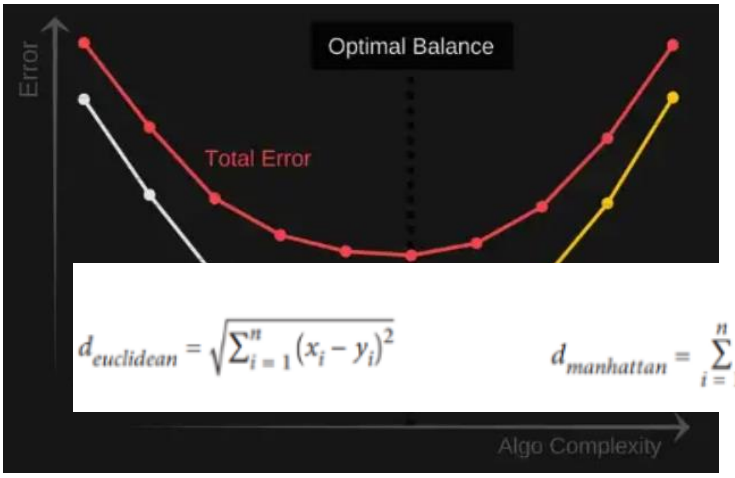
On left is bulls-eye diagram.

In supervised learning, underfitting happens when a model unable to capture the underlying pattern of the data.

overfitting happens when our model captures the noise along with the underlying pattern in data.

الگوریتم Tree Decision، الگوریتم پیچیده ای هست. پس bias پایینی دارد و ذاتا، درگیر مشکل over fitting که ناشی از variance بالا هست، می باشد. با انجام رای در الگوریتم RF میتوانیم از FIT شدن بیش از اندازه Tree Decision به روی noise ها جلوگیری کنیم و بنابراین به variance پایین تری برسیم. البته اندکی bias بیشتر میشود. ولی در کل performance بهتری خواهیم داشت و total error کاهش می یابد که مهم این است.(minimizeکردن total error).





$$d_{euclidean} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \qquad d_{manhattan} = \sum_{i=1}^{n}|x_i - y_i| \qquad d_{minkowski} = \left(\sum_{i=1}^{n}|x_i - y_i|^p\right)^{1/p}$$

Many machine learning algorithms like Gradient descent methods, KNN algorithm, linear and logistic

regression, etc. require data scaling to produce good results.
for knn with drop.

```
test_size: 0.7
{'normalize': 0, 'standardize': 1}
Logistic Regression
k-Nearest Neighbors
0.8136363636363637
{'normalize': 1, 'standardize': 0}
Logistic Regression
k-Nearest Neighbors
0.7954545454545455
{'normalize': 0, 'standardize': 0}
Logistic Regression
k-Nearest Neighbors
0.7787878787878787
test_size: 0.8
{'normalize': 0, 'standardize': 1}
Logistic Regression
k-Nearest Neighbors
0.8464285714285713
{'normalize': 1, 'standardize': 0}
Logistic Regression
k-Nearest Neighbors
0.8464285714285713
{'normalize': 0, 'standardize': 0}
Logistic Regression
k-Nearest Neighbors
0.8178571428571428
test_size: 0.9
{'normalize': 0, 'standardize': 1}
Logistic Regression
k-Nearest Neighbors
0.8333333333333334
{'normalize': 1, 'standardize': 0}
Logistic Regression
k-Nearest Neighbors
0.85
{'normalize': 0, 'standardize': 0}
Logistic Regression
k-Nearest Neighbors
0.95
```