Feature Importance and Feature Selection With XGBoost in Python

by **Jason Brownlee** on August 31, 2016 in **XGBoost**

Last Updated on August 21, 2019

A benefit of using ensembles of decision tree methods like gradient boosting is that they can automatically provide estimates of feature importance from a trained predictive model.

In this post you will discover how you can estimate the importance of features for a predictive modeling problem using the XGBoost library in Python.

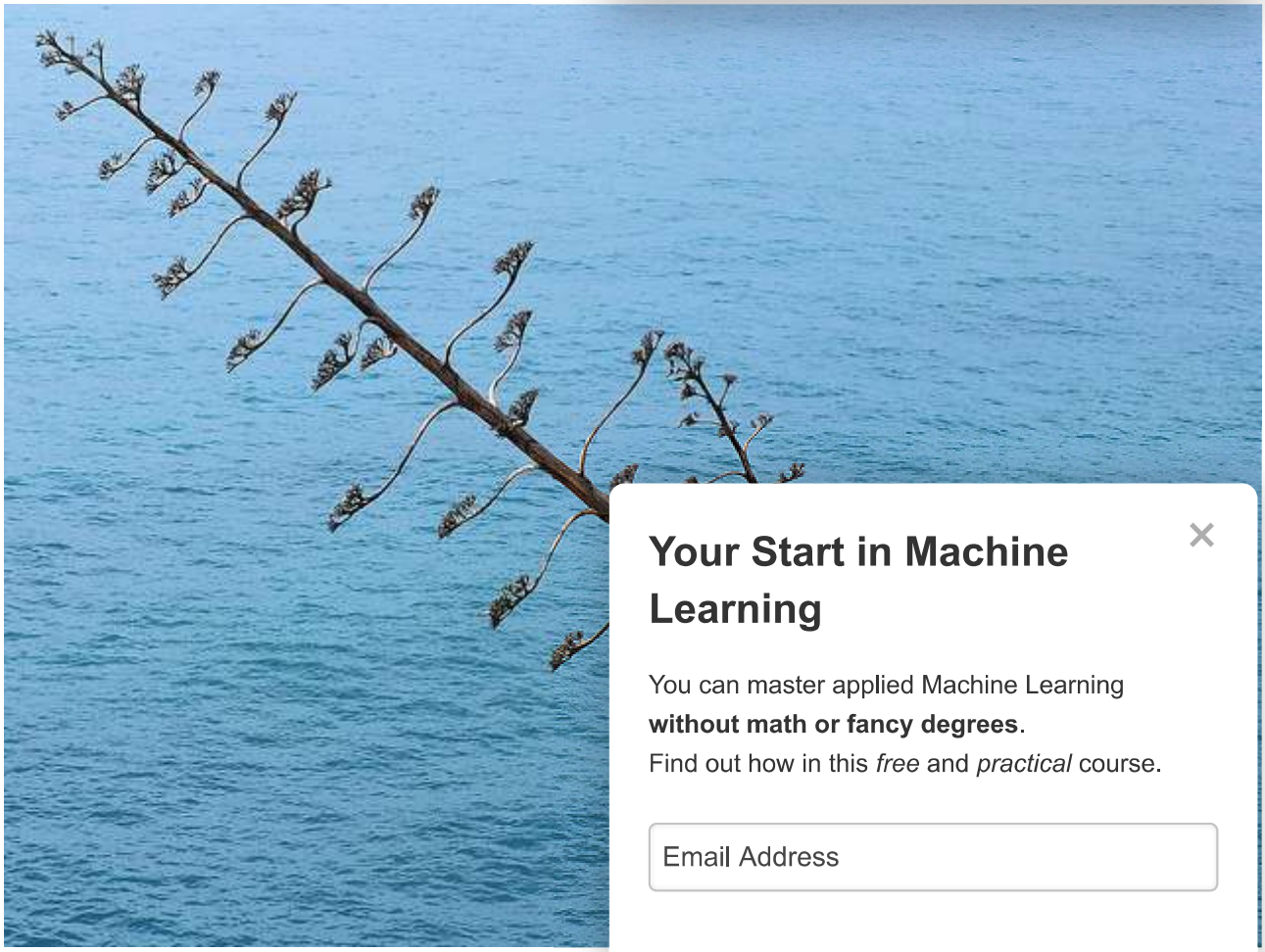
After reading this post you will know:

- How feature importance is calculated using the gradient boosting algorithm.
- How to plot feature importance in Python calculated by the XGBoost model.
- How to use feature importance calculated by XGBoost to perform feature selection.

Discover how to configure, fit, tune and evaluation gradient boosting models with XGBoost in my new book, with 15 step-by-step tutorial lessons, and full python code.

Let's get started.

- **Update Jan/2017**: Updated to reflect changes in scikit-learn API version 0.18.1.
- **Update March/2018**: Added alternate link to download the dataset as the original appears to have been taken down.



Feature Importance and Featur
Photo by Keith Rope

Feature Importance in Gradient Boosting

A benefit of using gradient boosting is that after the boosted trees are constructed, it is relatively straightforward to retrieve importance scores for each attribute.

Generally, importance provides a score that indicates how useful or valuable each feature was in the construction of the boosted decision trees within the model. The more an attribute is used to make key decisions with decision trees, the higher its *relative importance*.

This importance is calculated explicitly for each attribute in the dataset, allowing attributes to be ranked and compared to each other.

Importance is calculated for a single decision tree by the amount that each attribute split point improves the performance measure, weighted by the number of observations the node is responsible for. The performance measure may be the purity (Gini index) used to select the split points or another more specific error function.

The feature importances are then averaged across all of the the decision trees within the model.

For more technical information on how feature importance is calculated in boosted decision trees, see Section 10.13.1 "*Relative Importance of Predictor Variables*" of the book The Elements of Statistical Learning: Data Mining, Inference, and Prediction, p

Also, see Matthew Drury answer to the StackOver
Boosting" where he provides a very detailed and p

Manually Plot Feature Importa

A trained XGBoost model automatically calculates
problem.

These importance scores are available in the **featu**
model. For example, they can be printed directly a

```
print(model.feature_importances_)
```

We can plot these scores on a bar chart directly to get a visual indication of the relative importance of each feature in the dataset. For example:

```
# plot
pyplot.bar(range(len(model.feature_importances_)), model.feature_importances_)
pyplot.show()
```

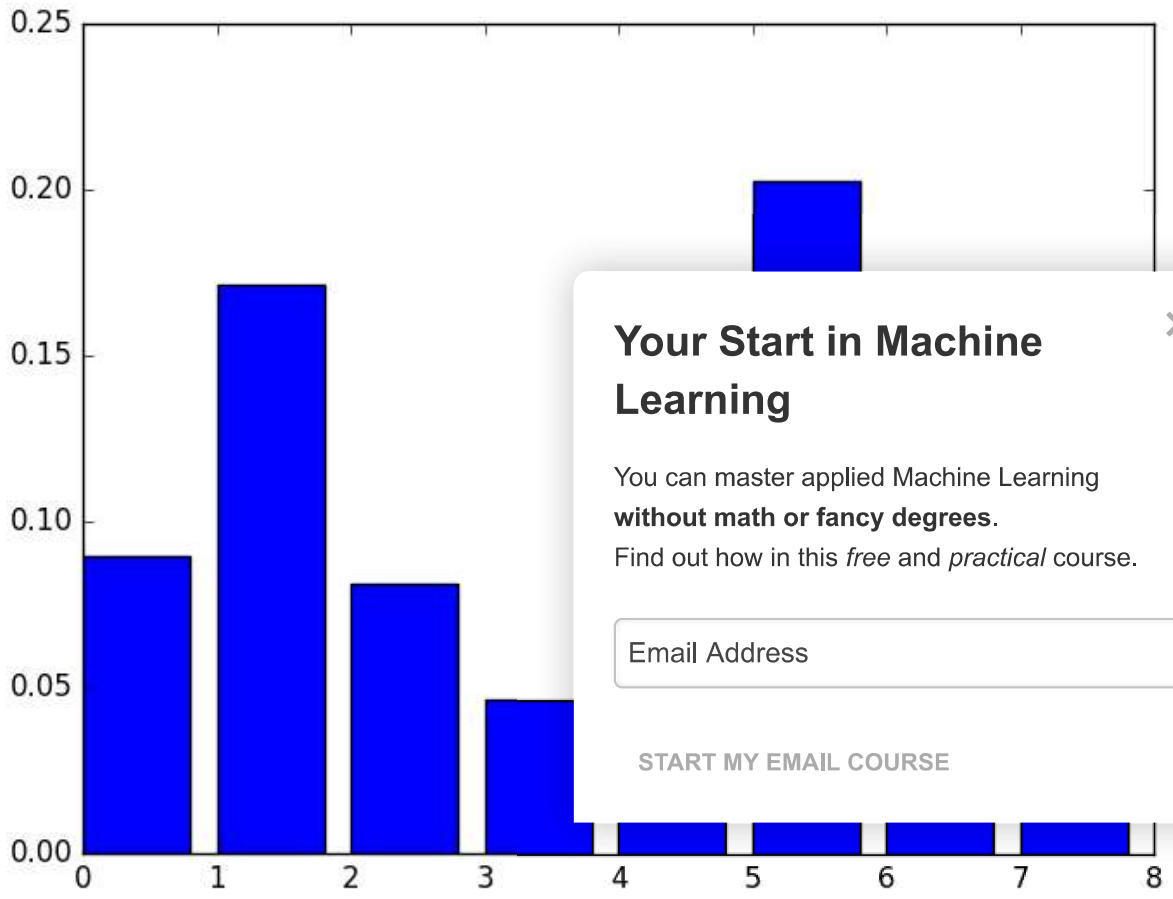
We can demonstrate this by training an XGBoost model on the Pima Indians onset of diabetes dataset and creating a bar chart from the calculated feature importances (update: download from here).

```
# plot feature importance manually
from numpy import loadtxt
from xgboost import XGBClassifier
from matplotlib import pyplot
# load data
dataset = loadtxt('pima-indians-diabetes.csv', delimiter=",")
# split data into X and y
X = dataset[:,0:8]
y = dataset[:,8]
# fit model no training data
model = XGBClassifier()
model.fit(X, y)
# feature importance
print(model.feature_importances_)
# plot
pyplot.bar(range(len(model.feature_importances_)), model.feature_importances_)
pyplot.show()
```

Running this example first outputs the importance

```
[ 0.089701    0.17109634  0.08139535  0.04651163  0.10465116  0.2026578 0.1627907   0.141196
```

We also get a bar chart of the relative importances.



Manual Bar Chart of XGBoost Feature Importance

A downside of this plot is that the features are ordered by their input index rather than their importance. We could sort the features before plotting.

Thankfully, there is a built in plot function to help us.

Using theBuilt-in XGBoost Feature Importance Plot

The XGBoost library provides a built-in function to plot features ordered by their importance.

The function is called **plot_importance()** and can be used as follows:

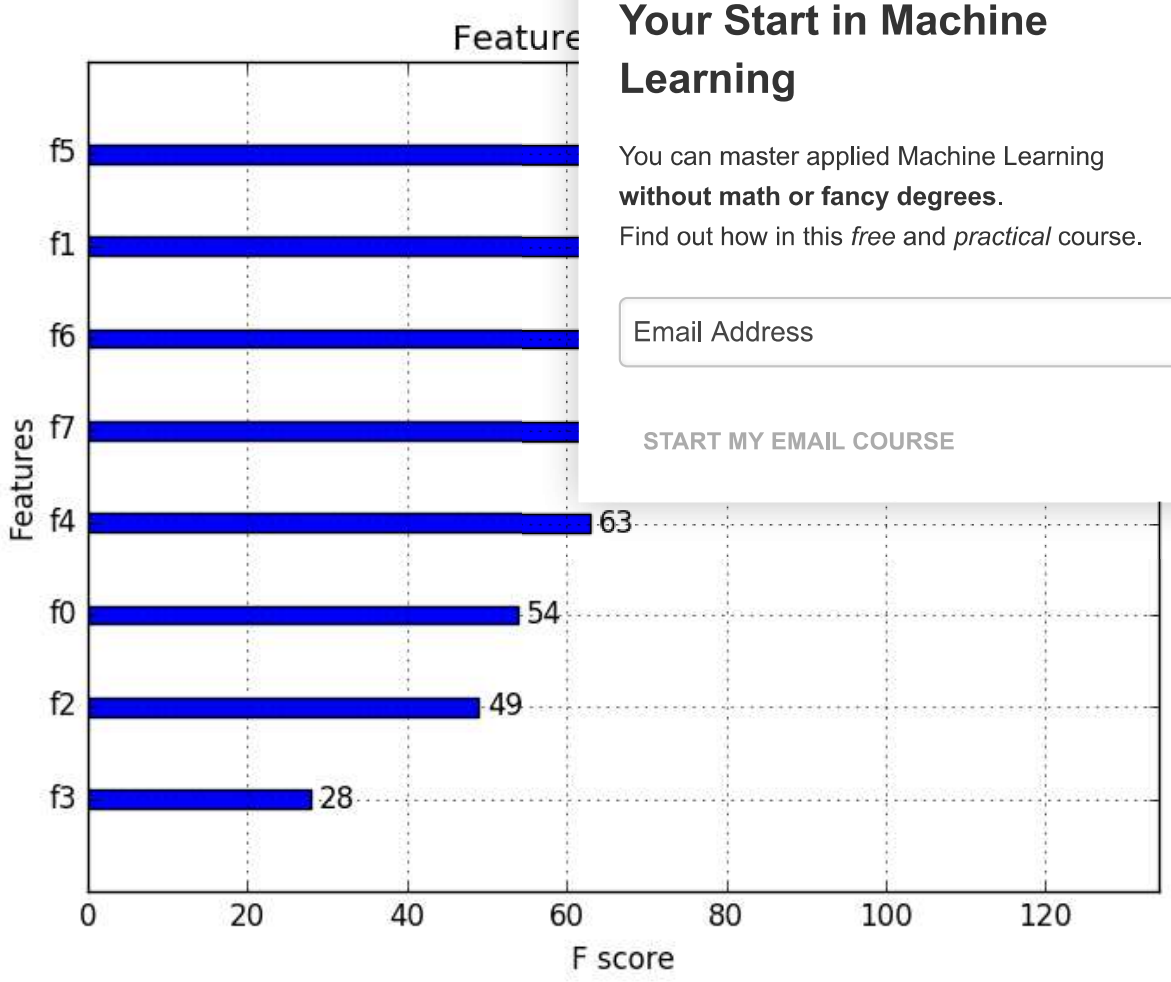
```
# plot feature importance
plot_importance(model)
pyplot.show()
```

For example, below is a complete code listing plotting the feature importance for the Pima Indians dataset using the built-in **plot_importance()** function.

```
# plot feature importance using built-in function
from numpy import loadtxt
from xgboost import XGBClassifier
```

```
from xgboost import plot_importance
from matplotlib import pyplot
# load data
dataset = loadtxt('pima-indians-diabetes.csv', delimiter=",")
# split data into X and y
X = dataset[:,0:8]
y = dataset[:,8]
# fit model no training data
model = XGBClassifier()
model.fit(X, y)
# plot feature importance
plot_importance(model)
pyplot.show()
```

Running the example gives us a more useful bar chart.



XGBoost Feature Importance Bar Chart

You can see that features are automatically named according to their index in the input array (X) from F0 to F7.

Manually mapping these indices to names in the problem description, we can see that the plot shows F5 (body mass index) has the highest importance and F3 (skin fold thickness) has the lowest importance.

Feature Selection with XGBoost Feature Importance Scores

Feature importance scores can be used for feature

This is done using the SelectFromModel class that takes a model and can transform a dataset into a subset with selected features.

This class can take a pre-trained model, such as one trained on the entire training dataset. It can then use a threshold to decide which features to select. This threshold is used when you call the **transform()** method on the **SelectFromModel** instance to consistently select the same features on the training dataset and the test dataset.

In the example below we first train and then evaluate an XGBoost model on the entire training dataset and test datasets respectively.

Using the feature importances calculated from the training dataset, we then wrap the model in a SelectFromModel instance. We use this to select f
the selected subset of features, then evaluate the
selection scheme.

For example:

```
# select features using threshold
selection = SelectFromModel(model, threshol
select_X_train = selection.transform(X_trai
# train model
selection_model = XGBClassifier()
selection_model.fit(select_X_train, y_train
# eval model
select_X_test = selection.transform(X_test)
y_pred = selection_model.predict(select_X_t
```


For interest, we can test multiple thresholds for selecting features by feature importance. Specifically, the feature importance of each input variable, essentially allowing us to test each subset of features by importance, starting with all features and ending with a subset with the most important feature.

The complete code listing is provided below.

```
# use feature importance for feature selection
from numpy import loadtxt
from numpy import sort
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.feature_selection import SelectFromModel
# load data
dataset = loadtxt('pima-indians-diabetes.csv', delimiter=",")
# split data into X and y
X = dataset[:,0:8]
Y = dataset[:,8]
# split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.33, random_state=7)
# fit model on all training data
model = XGBClassifier()
model.fit(X_train, y_train)
# make predictions for test data and evaluate
y_pred = model.predict(X_test)
predictions = [round(value) for value in y_pred]
accuracy = accuracy_score(y_test, predictions)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
# Fit model using each importance as a threshold
thresholds = sort(model.feature_importances_)
for thresh in thresholds:
    # select features using threshold
```

```
    selection = SelectFromModel(model, threshold=thresh, prefit=True)
    select_X_train = selection.transform(X_train)
    # train model
    selection_model = XGBClassifier()
    selection_model.fit(select_X_train, y_train)
    # eval model
    select_X_test = selection.transform(X_test)
    y_pred = selection_model.predict(select_X_test)
    predictions = [round(value) for value in y_pred]
    accuracy = accuracy_score(y_test, predictions)
    print("Thresh=%.3f, n=%d, Accuracy: %.2f%%" % (thresh, select_X_train.shape[1], accuracy
```

Running this example prints the following output:

```
Accuracy: 77.95%
Thresh=0.071, n=8, Accuracy: 77.95%
Thresh=0.073, n=7, Accuracy: 76.38%
Thresh=0.084, n=6, Accuracy: 77.56%
Thresh=0.090, n=5, Accuracy: 76.38%
Thresh=0.128, n=4, Accuracy: 76.38%
Thresh=0.160, n=3, Accuracy: 74.80%
Thresh=0.186, n=2, Accuracy: 71.65%
Thresh=0.208, n=1, Accuracy: 63.78%
```


We can see that the performance of the model ger
features.

On this problem there is a trade-off of features to t ss
complex model (fewer attributes such as n=4) and
from 77.95% down to 76.38%.

This is likely to be a wash on such a small dataset, but may be a more useful strategy on a larger dataset and using cross validation as the model evaluation scheme.

Summary

In this post you discovered how to access features and use importance in a trained XGBoost gradient boosting model.

Specifically, you learned:

- What feature importance is and generally how it is calculated in XGBoost.
- How to access and plot feature importance scores from an XGBoost model.
- How to use feature importance from an XGBoost model for feature selection.

Do you have any questions about feature importance in XGBoost or about this post? Ask your questions in the comments and I will do my best to answer them.

‹ How to Visualize Gradient Boosting Decision Trees With XGBoost in Python

Avoid Overfitting By Early Stopping With XGBoost In Python ›

108 Responses to *Feature Importance and Feature Selection With XGBoost in Python*

Trupti December 9, 2016 at 5:23 pm #

REPLY

Hi. I am running “select_X_train = selection.transform(X_train)” where x_train is the data with dependent variables in few rows.

The error I am getting is “select_X_train = selection.transform(X_train)”

Request your help.

Thanks!

Trupti December 9, 2016 at 5:28 pm #

sorry the error is “TypeError: only length-1 arrays can be converted to Python scalars”.

REPLY

Jason Brownlee December 10, 2016 at 8:04 am #

Check the shape of your X_train, e.g. print(X_train.shape)

You may need to reshape it into a matrix.

REPLY

sa January 5, 2017 at 3:44 pm #

I tried to select features for xgboost basec
since I am using gridsearch and pipeline, this error
select_X_train = selection.transform(X_train)
File “C:\Users\Markazi.co\Anaconda3\lib\site-packa
transform
mask = self.get_support()
File “C:\Users\MM.co\Anaconda3\lib\site-packages
get_support
mask = self._get_support_mask()
File “C:\Users\Markazi.co\Anaconda3\lib\site-packa
201, in _get_support_mask
scores = _get_feature_importances(estimator)
File “C:\Users\Markazi.co\Anaconda3\lib\site-packages\sklearn\feature_selection\from_model.py”, line 32, in _get_feature_importances
% estimator.__class__.__name__)
ValueError: The underlying estimator method has no `coef_` or `feature_importances_` attribute. Either pass a fitted estimator to SelectFromModel or call fit before calling transform.

regards,

REPLY

Jason Brownlee January 6, 2017 at 9:05 am #

Hi sa,

Consider trying the example without Pipelines first, get it working, then try adding in additional complexity.

REPLY

sa January 6, 2017 at 5:53 pm #

Hello Mr. Brownlee
Thanks

I already tried the example without Pipelines , and it works well. After adding pipeline, it could extract feature importance but after that it f

Best regards,

Johnn January 15, 2017 at 12:28 pm #

REPLY

Thanks for the post. I don't understand what's the meaning of "F-score" in the x-axis of the feature importance plot….. And what is the number next to each of the bar?

Jason Brownlee January 16, 2017 at 10:36 am #

REPLY

Hi Johnn,

You can learn more about the F1 score here:
https://en.wikipedia.org/wiki/F1_score

The number is a scaled importance, it really on

Gonçalo Abreu June 5, 2017 at 10:3

Hey Jason,

Are you sure the F score on the graph is re

I found this github page where the owner presents many ways to extract feature importance meaning from xgb. His explanation abou the F measure seems to have no relation to F1
https://github.com/Far0n/xgbfi

Jason Brownlee June 6, 2017 at 9:36 am #

REPLY

Importance scores are different from F scores. The above tutorial focuses on feature importance scores.

Soyoung Kim April 20, 2017 at 2:39 am #

REPLY

Hi Jason,

Your postings are always amazing for me to learn ML techniques!
Especially this XGBoost post really helped me work on my ongoing interview project.
The task is not for the Kaggle competition but for my technical interview! 🙂

I used your code to generate a feature importance ranking and some of the explanations you used to describe techniques.
You can find it here: https://www.kaggle.com/soyoungkim/two-sigma-connect-rental-listing-inquiries/rent-interest-classifier

Your Start in Machine Learning

I also put your link in the reference section.

Please let me know if it is not appropriate for me to use your code.

REPLY

Jason Brownlee April 20, 2017 at 9:31 am #

Well done.

As long as you cite the source, I am happy.

zttara April 27, 2017 at 3:42 pm #

Hi Jason,
I have some questions about feature importance.

I want to use the features that selected by XGBoos
I got confused on how to get the right scores of fea
to get the best model and obtain the correspondinc
right scores of features in the model?

Thanks a lot.

REPLY

Jason Brownlee April 28, 2017 at 7:36 am #

The scores are relative.

You can use them as a filter and select all features with a score above x, e.g. 0.5.

REPLY

max January 14, 2018 at 12:00 pm #

Hi Jason, I know that choosing a threshold (like 0.5) is always arbitray …but is there a rule of thumb for this?

thanks a lot.

REPLY

Jason Brownlee January 15, 2018 at 6:55 am #

Yes, start with 0.5, tune if needed.

Joe Butkovic August 26, 2019 at 10:58 pm #

Hi Jason,

Your Start in Machine Learning

Is there a score which should be discounted? For example, my highest score is 0.27, then 0.15, 0.13… Should I discount the model all together? Thanks!

Jason Brownlee August 27, 2019 at 6:46 am #

Scores are relative. Test different cut-off values on your specific dataset.

Shubham Jaiswal August 31, 2019 at 9:28 am #

One good way to not worr
CalibratedClassifierCV(clf, cv='pre

It kind of calibrated your classifier t

Jason Brownlee Septeml

Nice, thanks for sharing th

I also have a little more on the topi
https://machinelearningmastery.co

Omogbehin May 16, 2017 at 10:23 am # REPLY

Hello sir,

For the XGBoost feature selection, How do i change the Y axis to the names of my attributes. Kind regards sir.

Jason Brownlee May 17, 2017 at 8:23 am # REPLY

Great question, I'm not sure off-hand. You may need to use the xgboost API directly.

Franco Arda October 12, 2018 at 8:27 pm # REPLY

@Omogbehin, to get the Y labels automatically, you need to switch from arrays to Pandas dataframe. By doing so, you get automatically labeled Y and X.

```
column_names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
data = pd.read_csv("diabetes.csv", names = column_names)
X = data.iloc[:,0:8]
Y = data.iloc[:,8]
```

```
model = XGBClassifier()
model.fit(X, Y)

from xgboost import plot_importance
plot_importance(model)
plt.show()
```

Simone June 21, 2017 at 11:14 pm #

REPLY

Hi Jason,

Is it possible using “feature_importances_” in XGB

Jason Brownlee June 22, 2017 at 6:06 a

I’m not sure off the cuff, sorry.

Simone June 22, 2017 at 7:06 am #

Ok, I will try another method for fe

Thanks

Richard July 22, 2017 at 8:04 pm #

REPLY

Hello Jason, I use the XGBRegressor and want to do some feature selection. However, although the ‘plot_importance(model)’ command works, when I want to retreive the values using model.feature_importances_, it says ‘AttributeError: ‘XGBRegressor’ object has no attribute ‘feature_importances_’. Any hints how to retreive the feature importances for regression?

Jason Brownlee July 23, 2017 at 6:23 am #

REPLY

Sorry to hear that Richard. I’m not sure of the cause.

Long.Ye August 23, 2017 at 10:41 am #

REPLY

Hi Jason,

Do you know some methods to quality variable importance in RNN or LSTM? Could the XGBoost method be used in regression problems of RNN or LSTM? Thanks a lot.

Jason Brownlee August 23, 2017 at 4:23 pm #

REPLY

Perhaps, I have not tried.

Edward August 25, 2017 at 3:57 pm #

REPLY

Can you explain how the decision trees feature importance also works?

Biswajit September 9, 2017 at 10:36 pm #

Hi Jason while trying to fir my model in Xg

OSError: [WinError -529697949] Windows Error 0x

i am using 32 bit anaconda

import platform
platform.architecture()
('32bit', 'WindowsPE')

Please suggect how to get over this issue

Jason Brownlee September 11, 2017 at 12:01 pm #

REPLY

Sorry, I have not seen this error.

Perhaps you can post to stackoverflow?

kim tae in September 23, 2017 at 4:51 pm #

REPLY

Hi Jason.

SelectFromModel(model, threshold=thresh, prefit=True)

I wonder what prefit = true means in this section. I checked on the sklearn site, but I do not understand.

Jason Brownlee September 24, 2017 at 5:15 am #

REPLY

It specifies not to fit the model again, that we have already fit it prior.

Reed Guo January 19, 2018 at 2:14 am #

Hi, Jason

Can you get feature importance of artificial neural network?

If you can, how?

Thanks very much.

REPLY

Jason Brownlee January 19, 2018 at 6:35 am #

Perhaps, I have not seen this done.

Zhang January 25, 2018 at 11:41 pm #

Hi, Jason. I am doing a project with Stoch
and I think the ranking of feature importance can fe
machine can tell us which clinical features are mos
diseases. What I did is to predict the phenotypes o
using SGB in the training set, and then test the per
good (e.g., high accuracy and kappa), then I would
reasonable as machine can make good prediction
importance is the knowledge machine learns from
this knowledge to make good classification). Vice v
ranking of feature importance is bad or even wrong. In this case we cannot trust the 'knowledge' feed back by the machine. In other words, it wastes time to do feature selection in this case because the feature importance is not correct (either because of the poor data quality or the machine learning algorithm is not suitable). May I ask whether my thinking above is reasonable?
My second question is that I did not do feature selection to identify a subset of features as you did in your post. I just treat the few features on the top of the ranking list as the most important clinical features and then did classical analysis like t test to confirm these features are statistically different in different phenotypes. Can I still name it as feature selection or feature extraction? I am little bit confused about these terms. Thanks and I am waiting for your reply.

REPLY

Jason Brownlee January 26, 2018 at 5:42 am #

Sorry, I'm not sure I follow. Perhaps you can distil your question into one or two lines?

Yes, you could still call this feature selection.

REPLY

Zhang January 26, 2018 at 8:09 pm #

Thanks for your reply.

As you may know, stochastic gradient boosting (SGB) is a model with built-in feature selection, which is thought to be more efficient in feature selection tha

Your Start in Machine Learning

whether we can always trust the feature selected by SGB because the importance (relative influence) of the features are still provided by the model when the model has bad performance (e.g., very poor accuracy in testing). In this case, the model may be even wrong, so the selected features may be also wrong. So I would like to hear some comment from you regarding to this issue.

Thanks.

REPLY

Jason Brownlee January 27, 2018 at 5:57 am #

Perhaps compare models fit with different subsets of features to see if it is lifting skill.

Try using an ensemble of models fit on differen

Sa January 29, 2018 at 3:25 pm #

Hi, Jason.

Could you please let me know if the feature selecti
wrapper or embedded feature selection method?

Regards,

REPLY

Jason Brownlee January 30, 2018 at 9:47 am #

Here we are doing feature importance or feature scoring. It would be a filter.

REPLY

Youcai Wang February 2, 2018 at 9:37 am #

Hi Brownlee, if I have a dataset with 118 variables, but the target variable is in 116, and I want to use 6-115 and 117-118 variables as dependent variables, how can I modify the code X = dataset[:,0:8]
y = dataset[:,8]
to get X and Y?

I did not figure out this simple question. Please help

Thanks,

REPLY

Nick March 18, 2018 at 9:47 am #

Hi Jason,

Thanks for the tutorial.

Did you notice that the values of the importances were very different when you used
model.get_importances_ versus xgb.plot_importan

I used these two methods on a model I just trained and it looks like they are completely different. Moreover, the numpy array feature_importances do not directly correspond to the indexes that are returned from the plot_importance function.

In other words, these two methods give me qualitatively different results. Any idea why?

Jason Brownlee March 19, 2018 at 6:03 am #

REPLY

I have not noticed that. Perhaps post a ticket on the xgboost user group or on the project? Sounds like a fault?

Nick March 31, 2018 at 4:01 am #

There is a typo in my question:

It should be model.feature_importances, not m

Eran M April 3, 2018 at 9:19 pm #

Better importance estimation:

model.feature_importances_ uses the Booster.get_fscore() which uses Booster.get_score(importance_type='weight')

Which is an estimation to 'gain' (as of how many times all trees represented a certain feature).
I think it would be better to use Booster.get_score(importance_type='gain') to get a more precise evaluation of how important a feature is.
In general, it describes how good was it to split branches by that feature

Jason Brownlee April 4, 2018 at 6:12 am #

REPLY

Thanks for sharing.

John Markson November 22, 2018 at 12:42 am #

REPLY

Hi Jason

Thanks for all the awesome posts. Regarding the feature importance in Xgboost (or more generally gradient boosting trees), how do you feel about the SHAP? I am not sure if you already had any post discussing SHAP, but it *is definitely interesting* to people who need gradient boosting tree models for feature selections.

Thanks!

Your Start in Machine Learning

Jason Brownlee November 22, 2018 at 6:25 am #

REPLY

What is SHAP?

dasgupso May 24, 2018 at 10:01 pm #

REPLY

Hi Jason
I need to know the feature importance calculations by different methods like “weight”, “gain”, or “cover” etc. in Xgboost.

Please let me know how can we do it ? Can it be d
here (using Xgbclassifier).

Also what’s the default method which is giving varia
(model.feature_importances_).
I need to save importances for very large set of fea
etc. in Xgboost.

Jason Brownlee May 25, 2018 at 9:25 a

I’m not sure xgboost can present this, you might have to implement it yourself.

analysis.liu October 12, 2018 at 12:17 pm #

REPLY

you can use the following code:

model.get_booster().get_score(importance_type=’type’)

‘type’ can fill in ‘weight’ , ‘gain’ or ‘cover’.

Camel August 2, 2018 at 12:41 pm #

REPLY

Excuse me, I come across a problem when modeling with xgboost. Could I ask for your help? I use predict function to get a predict probability, but I get some prob which is below 0 or over 1. I’m wondering what’s my problem. Could you help me? Thank you very much.

Jason Brownlee August 2, 2018 at 2:10 pm #

REPLY

What is the problem exactly? Low probabilities?

Rocky September 9, 2018 at 5:20 am #

REPLY

Is it necessary to perform a gridsearch when comparing the performance of the model with different numbers of features? E.g if I wanted to see if a model with 8 features performed better than one with 4, would it be good practice to run a gridsearch with both?

Jason Brownlee September 9, 2018 at 6:01 am #

REPLY

It depends on how much time and resources you have and the goals of your project.

Perhaps a comparison of the same configuratio
good first step (w.g. without the grid search).

James September 26, 2018 at 12:33 pm #

How can we use let's say top 10 features
while initiating.

Jason Brownlee September 26, 2018 at

You must use feature selection methods to select the features you want to use. There is no best feature selection method, just different perspectives on what might be useful.

Sinan Ozdemir September 28, 2018 at 6:56 am #

REPLY

Hi Jason,

After reading your book, I was able to implement a model successfully. However, I have a few questions and I will appreciate if you provide feedback:

Q1 – In terms of feature selection, can we apply PCA (Principal Component Analysis), LDA (Linear Discriminant Analysis), or Kernel PCA when we use XGBOOST to determine the most important features?

Q2 – Do you think we should apply standard scaling after one hot encoding the categorical values? Again, some people say that this is not necessary in decision tree like models, but I would like to get your opinion.

Q3 – Do we need to be concerned with the dummy variable trap when we use XGBOOST? I couldn't find a good source about how XGBOOST handles the dummy variable trap meaning if it is necessary to drop a column.

As always I really appreciate your feedback.

Thank you.

Jason Brownlee September 28, 2018 at 2:58 pm #

REPLY

You can try dimensionality reduction methods, it really depends on the dataset and the configuration of the model as to whether they will be beneficial.

No real need to rescale data for xgboost. Standardizing might be useful for Gaussian variables. Test and see.

Dummy vars can be useful, especially if they expose a grouping of levels not obvious from the data (e.g. the addition of flag variables)

Sinan Ozdemir October 5, 2018 at 6

As always, thank you so much Ja

For people who are interested in my experi

Dimensionality reduction method didn't rea
was way better in my case.

Standardizing didn't really change neither t

Keeping dummy variable increased the acc
accuracy.

Thanks

Thanks

Jason Brownlee October 5, 2018 at 2:29 pm #

REPLY

Nice work!

Alvie December 12, 2018 at 3:07 pm #

REPLY

Hi Jason,

Thanks for your post. It is really helpful.

But I am still confused about "Importance is calculated for a single decision tree by the amount that each attribute split point improves the performance measure".

How to calculate the "amount that each attribute split point improves the performance measure"?

Jason Brownlee December 13, 2018 at 7:41 am #

REPLY

This is calculated as part of constructing each individual tree. The final importance scores are an average of these scores.

Yang Song December 27, 2018 at 7:09 pm #

REPLY

Hi Jason, I have encountered a problem when I try to reimplement the python trained xgboost model by c++. I built the same decision trees as the python trained(use the 'model.dump_model' function) but I got the different scores. I didn't know why and can't figure that,can you give me several tips? thanks!

Jason Brownlee December 28, 2018 at 5

Perhaps there was a difference in you

impossible for me to diagnose sorry.

Kamil February 12, 2019 at 7:14 am #

Hi,

when I'm running this code:

```
plot_importance(model)
pyplot.show()
```

I'm getting an error:

ValueError: tree must be Booster, XGBModel or dict instance

How can I deal with that?

Jason Brownlee February 12, 2019 at 8:09 am #

REPLY

Sorry, I have not seen that error, I have some suggestions here:
https://machinelearningmastery.com/faq/single-faq/why-does-the-code-in-the-tutorial-not-work-for-me

AAV March 16, 2019 at 9:16 am #

REPLY

Is there any way to get sign of the features to understand if the impact is positive or negative.

Jason Brownlee March 17, 2019 at 6:14

Not as far as I know, sorry.

Charles Brauer March 22, 2019 at 4:17 am #

REPLY

When I click on the link: “names in the problem description” I get a 404 error.
The “f1, f2.. ” names are not useful. I want the real column names.

Jason Brownlee March 22, 2019 at 8:37 am #

REPLY

Thanks, I have updated the link to:
https://github.com/jbrownlee/Datasets/blob/ma

Abhinav May 7, 2019 at 7:42 pm #

Hi Jason,

Thanks again for an awesome post. Just like there
selection using Random Forest.

Like – The categorical variable with high cardinality
(due to more number of splits)

And correlation is not visible in case of RF feature importance.

Do XGBoost have similar cons similar to Random Forest??

Jason Brownlee May 8, 2019 at 6:43 am #

REPLY

Yes, perhaps this post will help:
https://machinelearningmastery.com/configure-gradient-boosting-algorithm/

Constantine May 14, 2019 at 2:10 am #

REPLY

Hello!

Given feature importance is a very interesting property, I wanted to ask if this is a feature that can be found in other models, like Linear regression (along with its regularized partners), in Support Vector Regressors or Neural Networks, or if it is a concept solely defined solely for tree-based models. I ask because I am not sure whether I can consider eg Linear Regression’s coefficients as the analog for feature importance.

Thanks!

Jason Brownlee May 14, 2019 at 7:49 am #

REPLY

Yes, coefficient size in linear regression can be a sign of importance.

SVM, less so.

Constantine May 16, 2019 at 4:39 am #

REPLY

Many thanks!

Jason Brownlee May 16, 2019 at 6:36 a

You're welcome.

Hiro May 19, 2019 at 1:38 am #

Thanks for all of your posts. I use your blo
I have a question. If you had a large number of fea
dataset with over 1,000 features but not all of them
working on. Should I reduce the number of features

Jason Brownlee May 19, 2019 at 8:05 am #

REPLY

Try modeling with all features and compare results to models fit on subsets of selected features to see if it improves performance.

Jonathan May 21, 2019 at 4:08 am #

REPLY

Hi Jason,

Does multicollinearity affect feature importance for boosted regression trees? If so, how would you suggest to treat this problem?

Thanks!

Jason Brownlee May 21, 2019 at 6:40 am #

REPLY

Probably not.

Try modeling with an without the colinear features and compare results

Grzegorz Kępisty July 16, 2019 at 5:03 pm #

REPLY

Hello Jason,

Concerning default feature importance in similar method from sklearn (Random Forest) I recommend meaningful article :
https://explained.ai/rf-importance/
The authors show that the default feature importance implementation using Gini is biased.

I observed this kind of bias several times, that is overestimation of importance of artificial random variables added to data sets. For this issue – so called – permutation importance was a solution at a cost of longer computation.

However, there are other methods like “drop-col im
while working with production data, I observed that
in its tail – depending which method of 2 above I ap

This is somehow confusing and now I am cautious
Do you have some experience in this field or some

Best regards!

Jason Brownlee July 17, 2019 at 8:20 ar

Thanks for sharing.

My best advice is to use importance as a suggestion but remain skeptical. Test many methods, many subsets, make features earn the use in the model with hard evidence.

new_to_modelling July 17, 2019 at 1:08 am #

REPLY

My data only has 6 columns, where i want to predict one of those columns so remaining 5. Out of which 2 are categorical variable and 3 are numerical variable. So, i used https://scikit-learn.org/stable/auto_examples/compose/plot_column_transformer_mixed_types.html to workout a mixed data type issues. But when i the feature_importance size does not match with the original number of columns? The size of feature_importances_ array is 918

I mean its generating extra feature or is it creating a feature value for one_hot_encoding of the categorical variable.

I checked my data has 1665 unique brand values. So, its not the same as feature_importances_ array size

Jason Brownlee July 17, 2019 at 8:28 am #

REPLY

Performing feature selection on the categorical data might be confusing as it is probably one hot encoded.

Perhaps create a subset of the data with just the numerical features and perform feature selection on that?

new_to_modelling July 17, 2019 at 6:45 pm #

REPLY

Thanks, but i found it was working once i tried dummies in place of the above mentioned column transformer approach seems like during transformation there is some loss of information when the xgboost booster picks up the feature names

Jason Brownlee July 18, 2019

Glad to hear it.

Mike Sishi July 28, 2019 at 4:22 am #

Interesting article, thanks a lot!!

Ho can I reverse-engineer a Decision Tree? That is
feature variables adjust themselves.

Basically, I want to set a target variable value and g
yield the target variable value.

Jason Brownlee July 28, 2019 at 6:49 am #

REPLY

Reverse ML/predictive modeling is very hard if not entirely intractable.

You could turn one tree into rules and do this and give many “results”.

It would might not make sense for an ensemble of trees.

Mike Sishi July 28, 2019 at 6:27 pm #

REPLY

Hi Jason,

Thanks for your prompt response. I will will try to work on the solution and let you know how it goes.

Kind Regards

Jason Brownlee July 29, 2019 at 6:11 am #

REPLY

Good luck!

REPLY

Abdoul August 17, 2019 at 3:00 am #

How to extract the n best attributs at the end?

REPLY

Jason Brownlee August 17, 2019 at 5:58 am #

I give an example in the above tutorial

Robert Feyerharm August 28, 2019 at 11:49

Thanks Jason, very helpful!

Is there a way to determine if a feature has a net p
variable?

Jason Brownlee August 29, 2019 at 6:12

Yes, you can calculate the correlation between them.

REPLY

Roger September 6, 2019 at 10:10 am #

precision: 51.85%
Thresh=0.030, n=10, precision: 46.81%
Thresh=0.031, n=9, precision: 50.00%
Thresh=0.032, n=8, precision: 47.83%
Thresh=0.033, n=7, precision: 51.11%
Thresh=0.035, n=6, precision: 48.78%
Thresh=0.041, n=5, precision: 41.86%
Thresh=0.042, n=4, precision: 58.62%
Thresh=0.043, n=3, precision: 68.97%
Thresh=0.045, n=2, precision: 62.96%
Thresh=0.059, n=1, precision: 0.00%

Hi Jason, Thank you for your post, and I am so happy to read this kind of useful ML articles. I have a question: the above output is from my example. As you can see, when thresh = 0.043 and n = 3, the precision dramatically goes up. So, I want to take a closer look at that thresh and wants to find out the names and corresponding feature importances of those 3 features. How can I achieve this goal?

Jason Brownlee September 6, 2019 at 1:57 pm #

REPLY

Each feature has a unique index of the column in the dataset from 0 to n. If you know the names of the columns, you can map the column index to names.

You can then do this in Python to automate it.

I hope that helps.

Ralph September 21, 2019 at 5:45 pm #

REPLY

Hi! I am using instead the xgb.train comm
faster. By the way you have any idea why, and if it
XGBClassifier (might be related to the number of t

Anyway, you have any idea of how to get importan

Many thanks

Jason Brownlee September 22, 2019 at

Are you sure it is faster? It should be i

abstract September 25, 2019 at 1:56 pm #

REPLY

Any reason why the Accuracy has increased from 76.38 at n=7 to 77.56 at n=6 ?

Jason Brownlee September 26, 2019 at 6:27 am #

REPLY

Perhaps the change in inputs or perhaps the stochastic nature of the learning algorithm.

A fair comparison would use repeated k-fold cross validation and perhaps a significance test.

Maria September 27, 2019 at 11:50 pm #

REPLY

Hello Jason,

I work on an imbalanced dataset for annomaly detection in machines. I have 590 features and 1567 observations. I tried this approach for reducing the number of features since I noticed there was multicollinearity, however, there is no important shift in the results for my precision and recall and sometimes the results get really weird. I was wondering what could that be an indication of?
Here are the results of the features selection

```
Thresh=0.000, n=211, f1_score: 5.71%
precision_score: 50.00%
recall_score: 3.03%
accuracy_score: 91.22%
Thresh=0.000, n=210, f1_score: 5.71%
precision_score: 50.00%
recall_score: 3.03%
accuracy_score: 91.22%
Thresh=0.000, n=209, f1_score: 5.71%
precision_score: 50.00%
recall_score: 3.03%
accuracy_score: 91.22%
Thresh=0.000, n=208, f1_score: 5.71%
precision_score: 50.00%
recall_score: 3.03%
accuracy_score: 91.22%
Thresh=0.000, n=207, f1_score: 5.71%
precision_score: 50.00%
recall_score: 3.03%
accuracy_score: 91.22%
.
.
.
Thresh=0.006, n=55, f1_score: 11.11%
precision_score: 66.67%
recall_score: 6.06%
accuracy_score: 91.49%
Thresh=0.006, n=54, f1_score: 5.88%
precision_score: 100.00%
recall_score: 3.03%
accuracy_score: 91.49%
Thresh=0.007, n=53, f1_score: 5.88%
precision_score: 100.00%
recall_score: 3.03%
accuracy_score: 91.49%
Thresh=0.007, n=52, f1_score: 5.88%
precision_score: 100.00%
recall_score: 3.03%
accuracy_score: 91.49%
.
.
.
Thresh=0.007, n=47, f1_score: 0.00%
precision_score: 0.00%
recall_score: 0.00%
accuracy_score: 91.22%
```

UndefinedMetricWarning: F-score is ill-defined and being set to 0.0 due to no predicted samples. 'precision', 'predicted', average, warn_for)

Your Start in Machine Learning

Precision is ill-defined and being set to 0.0 due to no predicted samples.
'precision', 'predicted', average, warn_for)

Jason Brownlee September 28, 2019 at 6:20 am #

REPLY

Intersting, I'm not sure.

You may need to dig into the specifics of the data to what is going on. If you're using CV, then perhaps some folds don't have examples of the target class – use stratified CV.

Maria September 28, 2019 at 8:41 pm #

You're right. I have 104 exemples
Would you suggest oversampling in that ca

Jason Brownlee September 2

Perhaps test it and see.

More ideas here:
https://machinelearningmastery.com/ta
learning-dataset/

Leave a Reply

Name (required)

Email (will not be published) (required)

Website

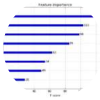
Your Start in Machine Learning

SUBMIT COMMENT

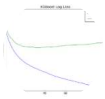
Welcome! I'm **Jason Brownlee** PhD and I help developers get results with machine learning. Read More

Picked for you:

How to Develop Your First XGBoost Model in Py



Feature Importance and Feature Selection With



Avoid Overfitting By Early Stopping With XGBoo

How to Visualize Gradient Boosting Decision Tr

A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning

Loving the Tutorials?

The XGBoost With Python EBook is where I keep the ***Really Good*** stuff.

SEE WHAT'S INSIDE

Your Start in Machine Learning