**Feature Importance**

**Feature importance** refers to techniques that assign a score to input features based on how useful they are at predicting a target variable.

There are many types and sources of feature importance scores, although popular examples include statistical correlation scores, coefficients calculated as part of linear models, decision trees, and permutation importance scores.

Feature importance scores play an important role in a predictive modeling project, including providing insight into the data, insight into the model, and the basis for dimensionality reduction and feature selection that can improve the efficiency and effectiveness of a predictive model on the problem.

In this tutorial, you will discover feature importance scores for machine learning in python

After completing this tutorial, you will know:

* The role of feature importance in a predictive modeling problem.
* How to calculate and review feature importance from linear models and decision trees.
* How to calculate and review permutation feature importance scores.

## Tutorial Overview

This tutorial is divided into six parts; they are:

1. Feature Importance
2. Preparation
   1. Test Datasets
3. Coefficients as Feature Importance
   1. Linear Regression Feature Importance
   2. Logistic Regression Feature Importance
4. Decision Tree Feature Importance
   1. CART Feature Importance
   2. Random Forest Feature Importance
   3. XGBoost Feature Importance
5. Permutation Feature Importance
   1. Permutation Feature Importance for Regression
   2. Permutation Feature Importance for Classification
6. Feature Selection with Importance

## **Feature Importance**

Feature importance refers to a class of techniques for assigning scores to input features to a predictive model that indicates the relative importance of each feature when making a prediction.

Feature importance scores can be calculated for problems that involve predicting a numerical value, called regression, and those problems that involve predicting a class label, called classification.

The scores are useful and can be used in a range of situations in a predictive modeling problem, such as:

* Better understanding the data.
* Better understanding a model.
* Reducing the number of input features.

**Feature importance scores can provide insight into the dataset**. The relative scores can highlight which features may be most relevant to the target, and the converse, which features are the least relevant. This may be interpreted by a domain expert and could be used as the basis for gathering more or different data.

**Feature importance scores can provide insight into the model**. Most importance scores are calculated by a predictive model that has been fit on the dataset. Inspecting the importance score provides insight into that specific model and which features are the most important and least important to the model when making a prediction. This is a type of model interpretation that can be performed for those models that support it.

**Feature importance can be used to improve a predictive model**. This can be achieved by using the importance scores to select those features to delete (lowest scores) or those features to keep (highest scores). This is a type of feature selection and can simplify the problem that is being modeled, speed up the modeling process (deleting features is called dimensionality reduction), and in some cases, improve the performance of the model.

*Often, we desire to quantify the strength of the relationship between the predictors and the outcome. […] Ranking predictors in this manner can be very useful when sifting through large amounts of data.*

Feature importance scores can be fed to a wrapper model, such as the [SelectFromModel](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectFromModel.html) class, to perform feature selection.

There are many ways to calculate feature importance scores and many models that can be used for this purpose.

Perhaps the simplest way is to calculate simple coefficient statistics between each feature and the target variable.

In this tutorial, we will look at three main types of more advanced feature importance; they are:

* Feature importance from model coefficients.
* Feature importance from decision trees.
* Feature importance from permutation testing.

Let’s take a closer look at each.

### **Test Datasets**

Next, let’s define some test datasets that we can use as the basis for demonstrating and exploring feature importance scores.

Each test problem has five important and five unimportant features, and it may be interesting to see which methods are consistent at finding or differentiating the features based on their importance.

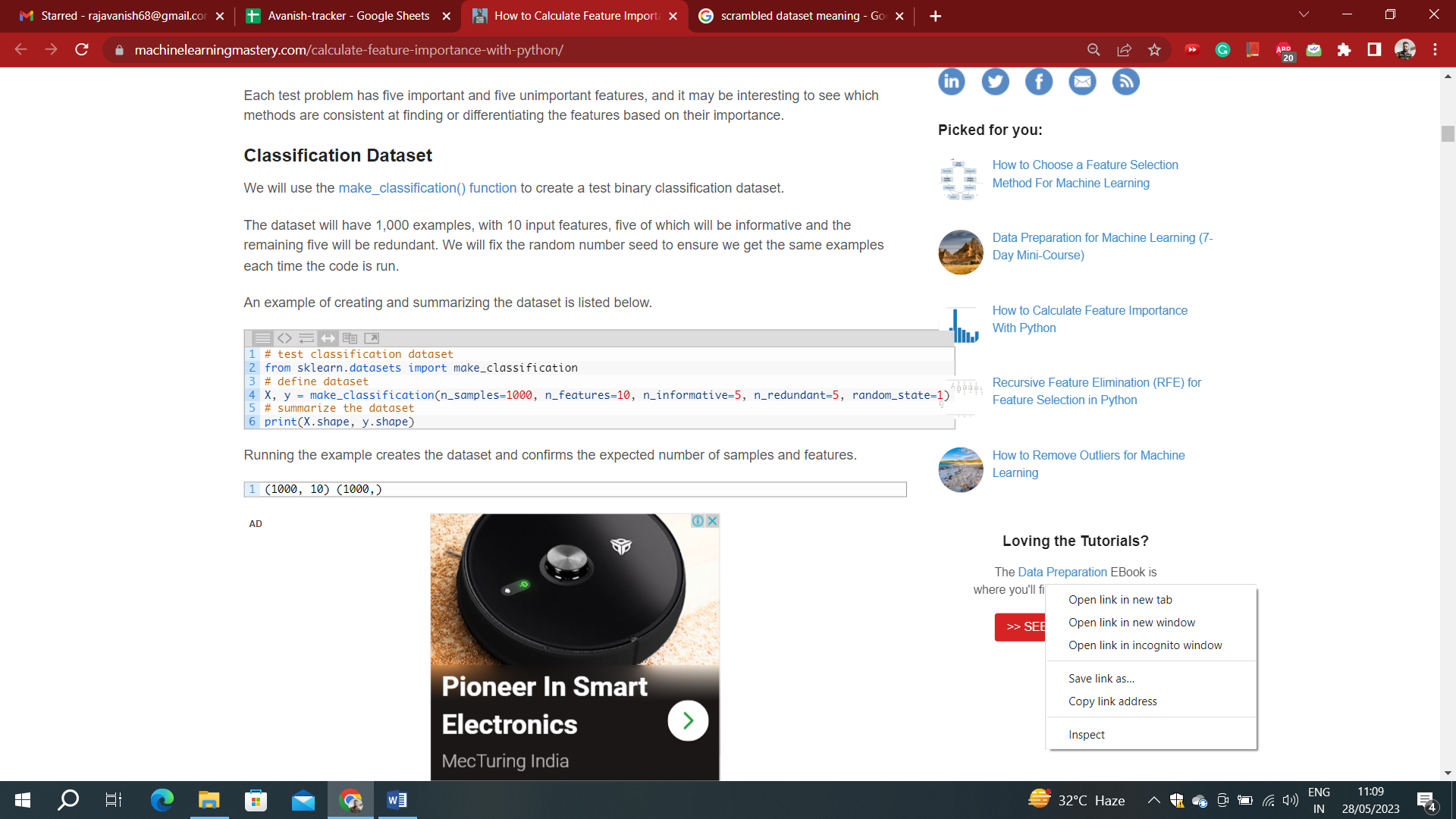
**AD**

### **Classification Dataset**

We will use the [make\_classification() function](https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html) to create a test binary classification dataset.

The dataset will have 1,000 examples, with 10 input features, five of which will be informative and the remaining five will be redundant. We will fix the random number seed to ensure we get the same examples each time the code is run.

An example of creating and summarizing the dataset is listed below.



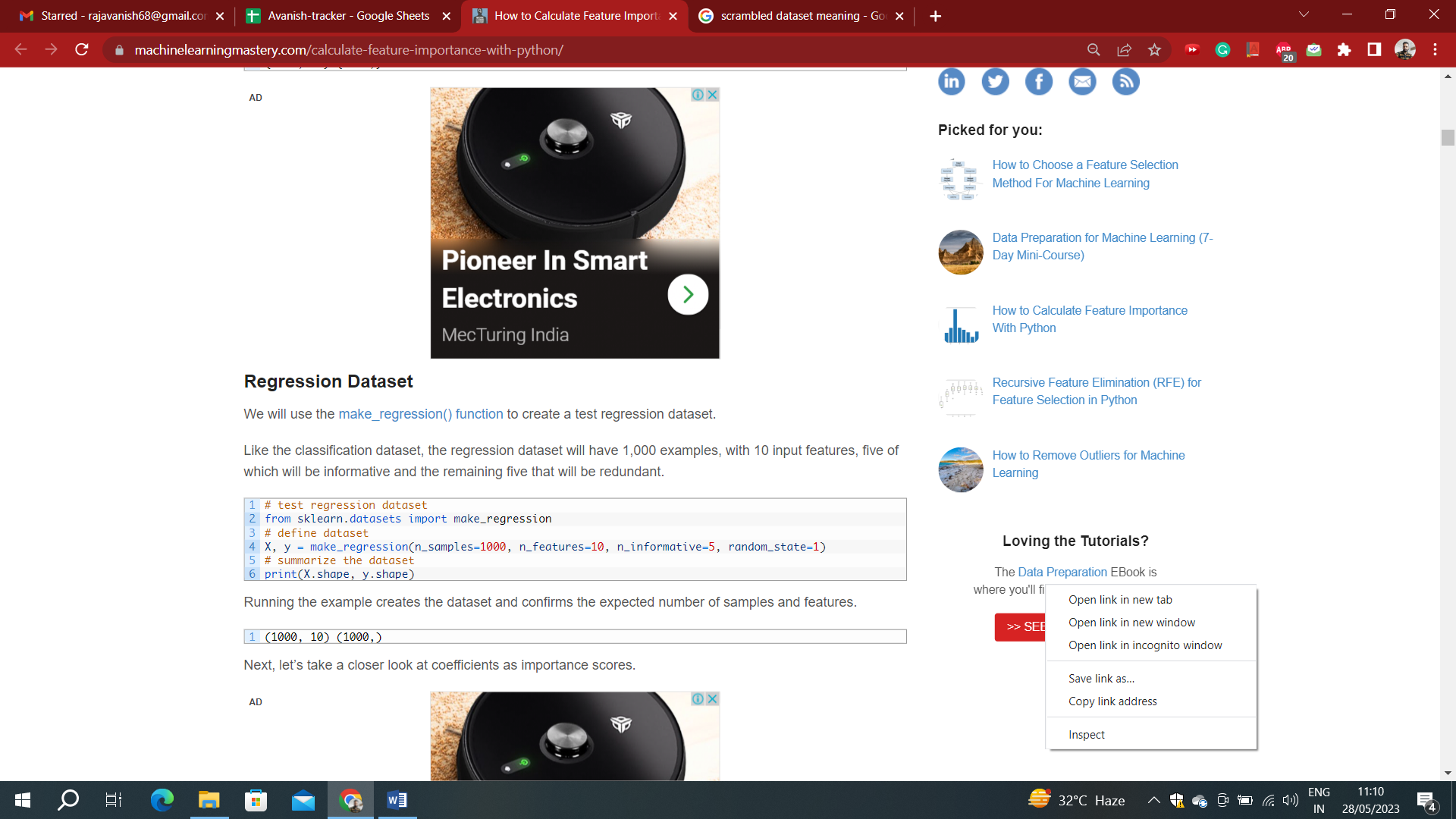
Run

(1000, 10) (1000,)

### **Regression Dataset**

We will use the [make\_regression() function](https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_regression.html) to create a test regression dataset.

Like the classification dataset, the regression dataset will have 1,000 examples, with 10 input features, five of which will be informative and the remaining five that will be redundant.



Run

(1000, 10) (1000,)

Next, let’s take a closer look at coefficients as importance scores.

## **Coefficients as Feature Importance**

Linear machine learning algorithms fit a model where the prediction is the weighted sum of the input values.

Examples include linear regression, logistic regression, and extensions that add regularization, such as ridge regression and the elastic net.

All of these algorithms find a set of coefficients to use in the weighted sum in order to make a prediction. These coefficients can be used directly as a crude type of feature importance score.

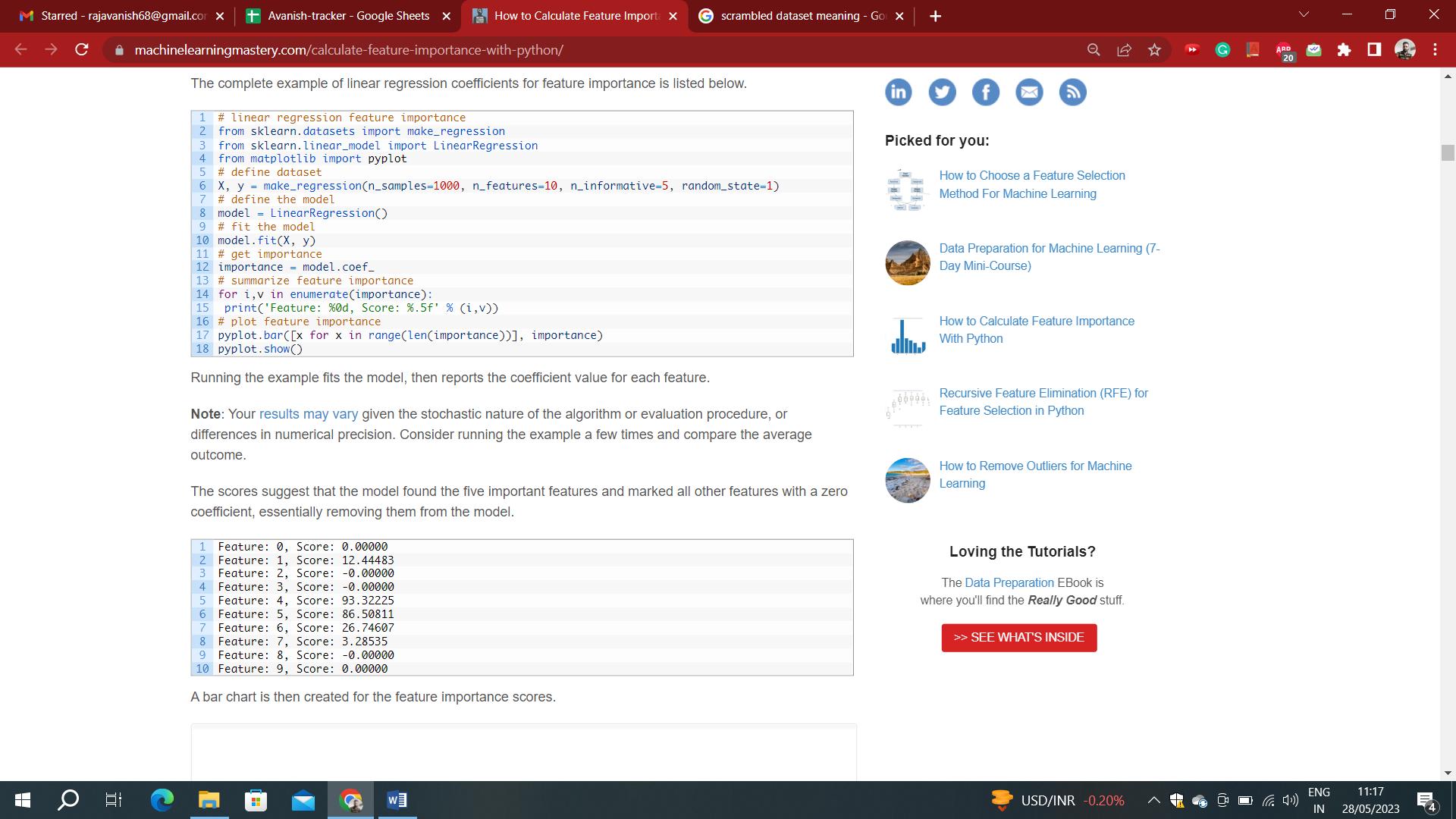
Let’s take a closer look at using coefficients as feature importance for classification and regression. We will fit a model on the dataset to find the coefficients, then summarize the importance scores for each input feature and finally create a bar chart to get an idea of the relative importance of the features.

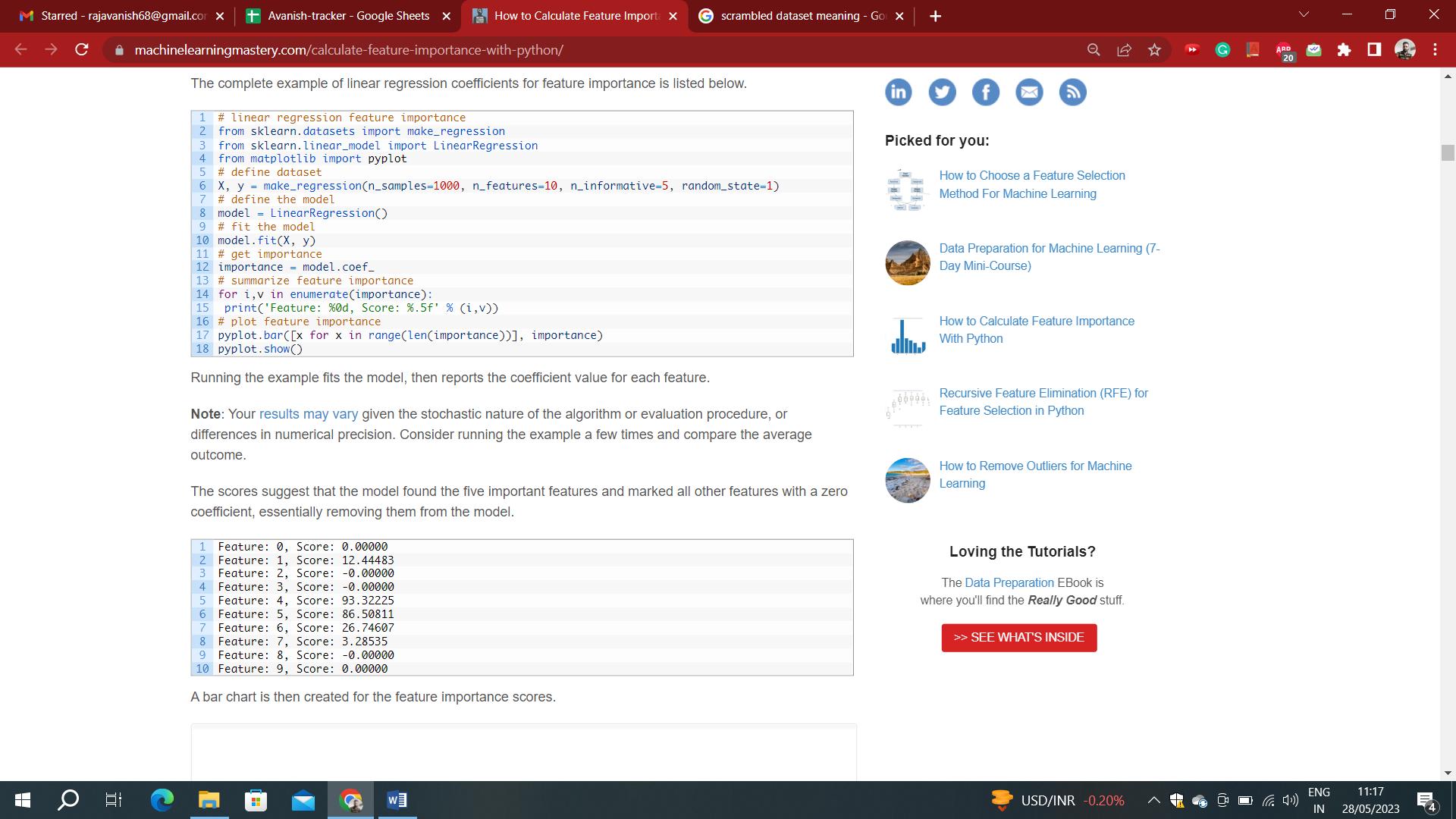
### **Linear Regression Feature Importance**

We can fit a [LinearRegression](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html) model on the regression dataset and retrieve the coeff\_ property that contains the coefficients found for each input variable.

These coefficients can provide the basis for a crude feature importance score. This assumes that the input variables have the same scale or have been scaled prior to fitting a model.

The complete example of linear regression coefficients for feature importance is listed below.

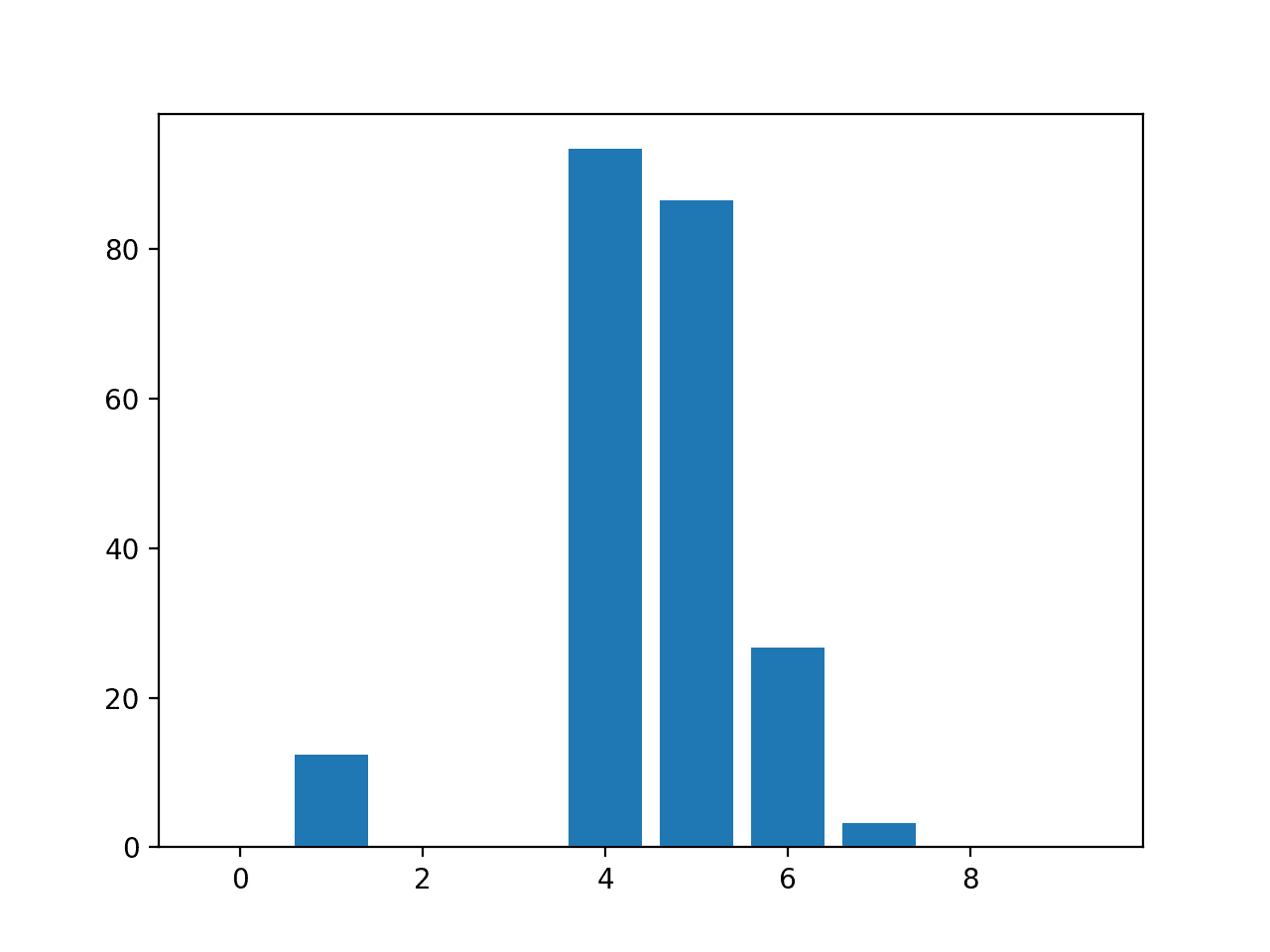




Running the example fits the model, then reports the coefficient value for each feature.

The scores suggest that the model found the five important features and marked all other features with a zero coefficient, essentially removing them from the model.

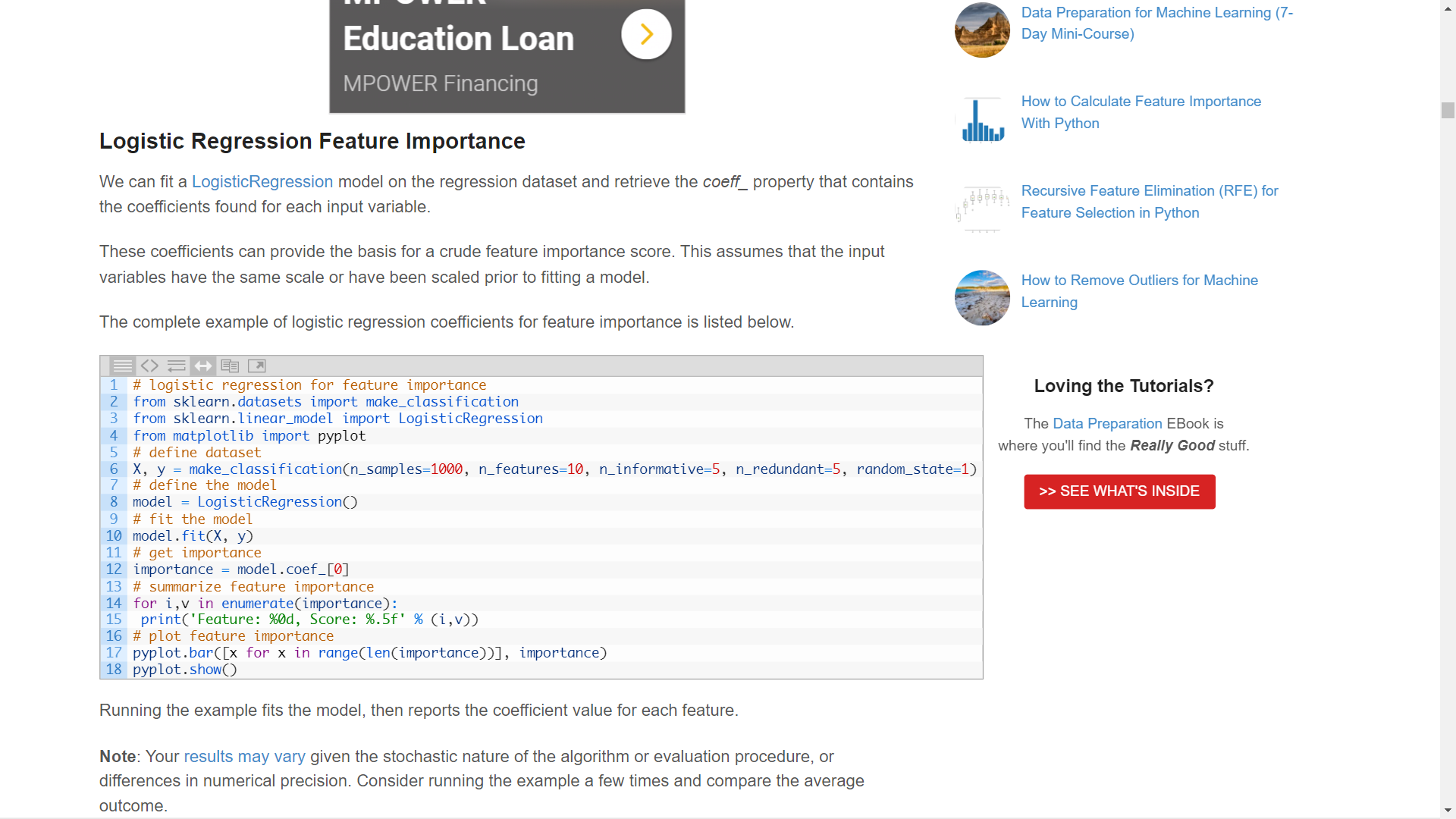
A bar chart is then created for the feature importance scores.

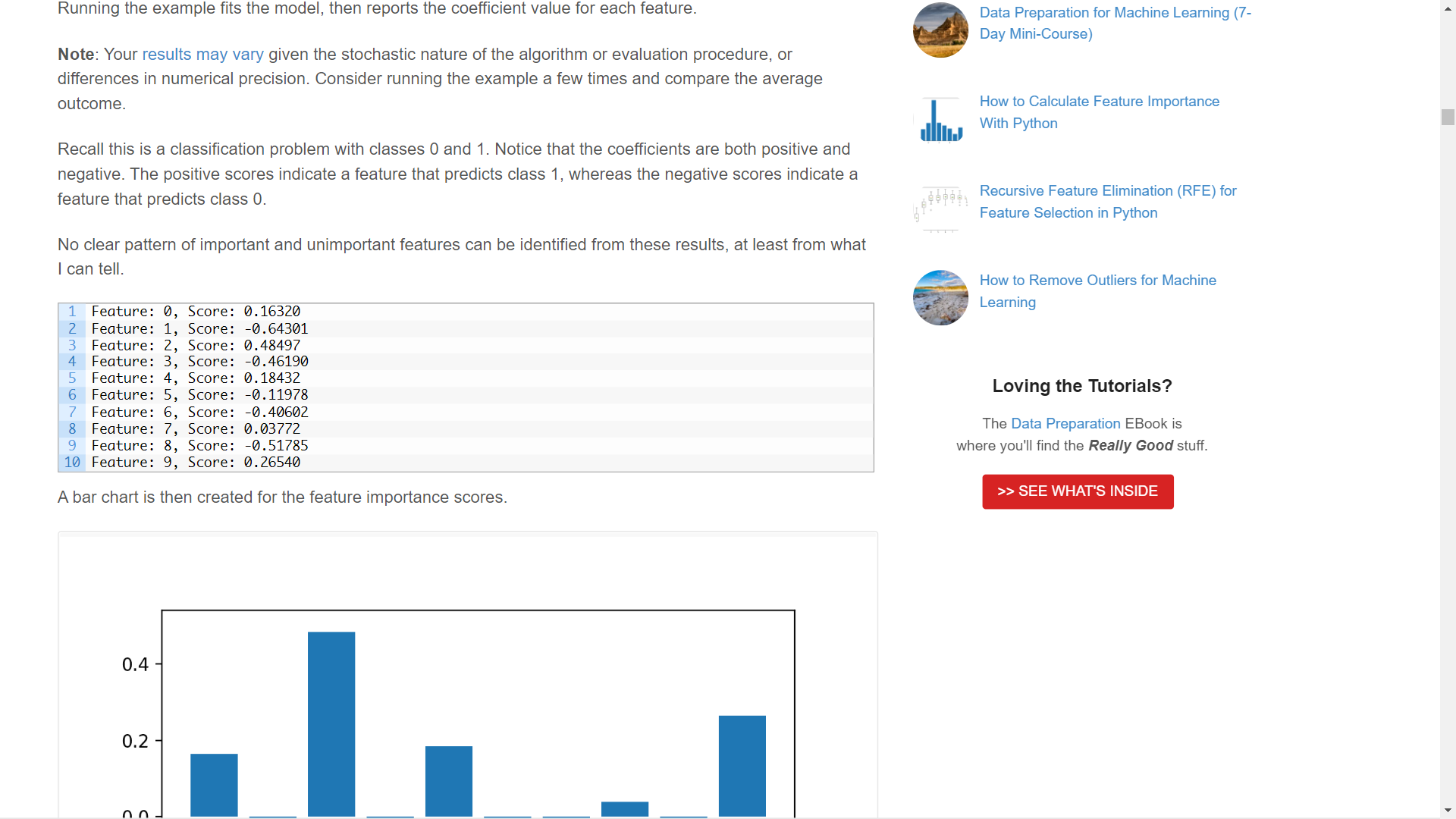


Bar Chart of Linear Regression Coefficients as Feature Importance Scores

**Note:** This approach may also be used with Lasso, [Ridge](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html) and [ElasticNet](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html) models.

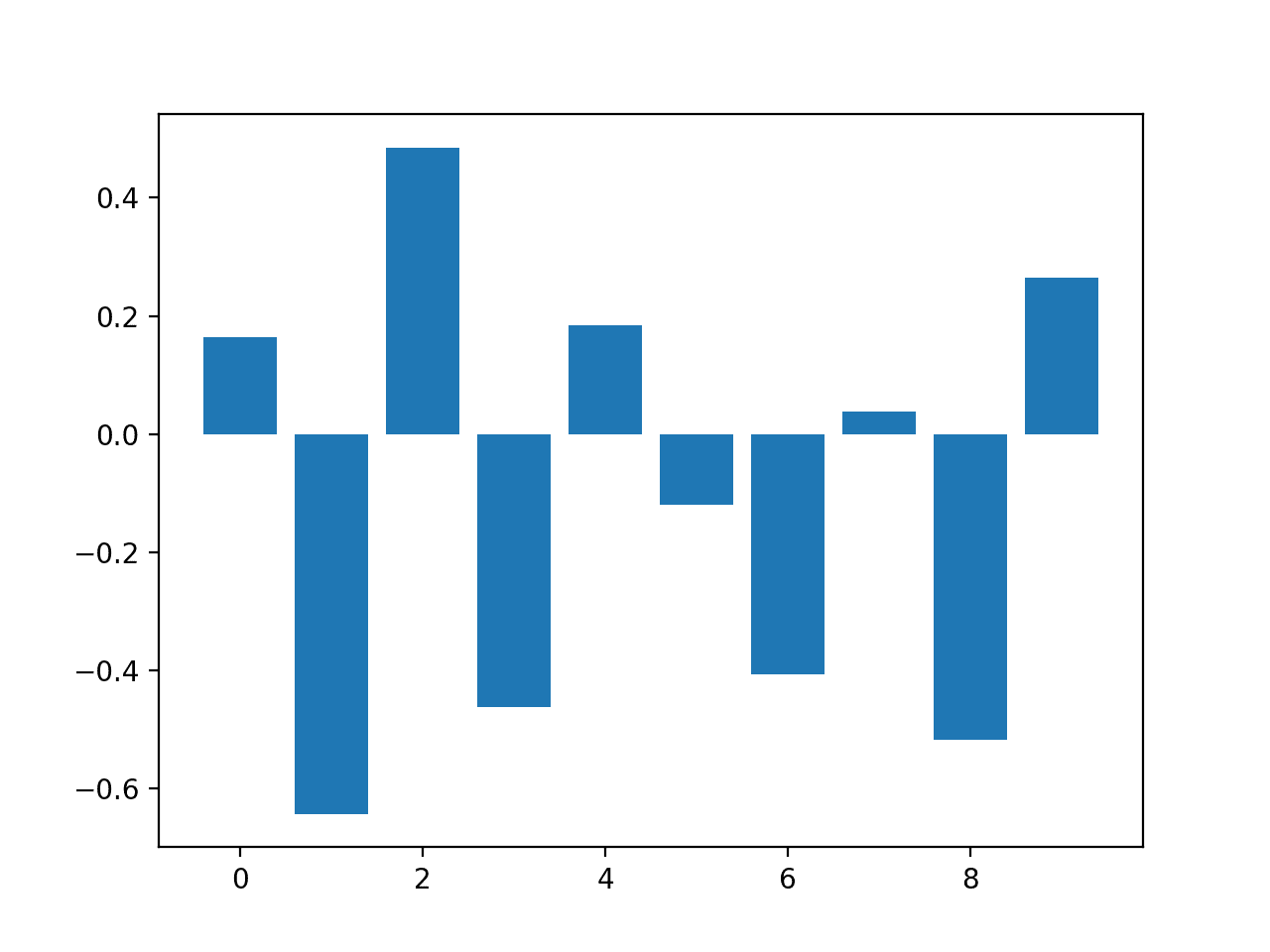
### **Logistic Regression Feature Importance**





Recall this is a classification problem with classes 0 and 1. Notice that the coefficients are both positive and negative. The positive scores indicate a feature that predicts class 1, whereas the negative scores indicate a feature that predicts class 0.

No clear pattern of important and unimportant features can be identified from these results, at least from A bar chart is then created for the feature importance scores.



Bar Chart of Logistic Regression Coefficients as Feature Importance Scores

Now that we have seen the use of coefficients as importance scores, let’s look at the more common example of decision-tree-based importance scores.

## **Decision Tree Feature Importance**

Decision tree algorithms like [classification and regression trees](https://machinelearningmastery.com/implement-decision-tree-algorithm-scratch-python/) (CART) offer importance scores based on the reduction in the criterion used to select split points, like Gini or entropy.

This same approach can be used for ensembles of decision trees, such as the random forest and stochastic gradient boosting algorithms.

### **CART Feature Importance**

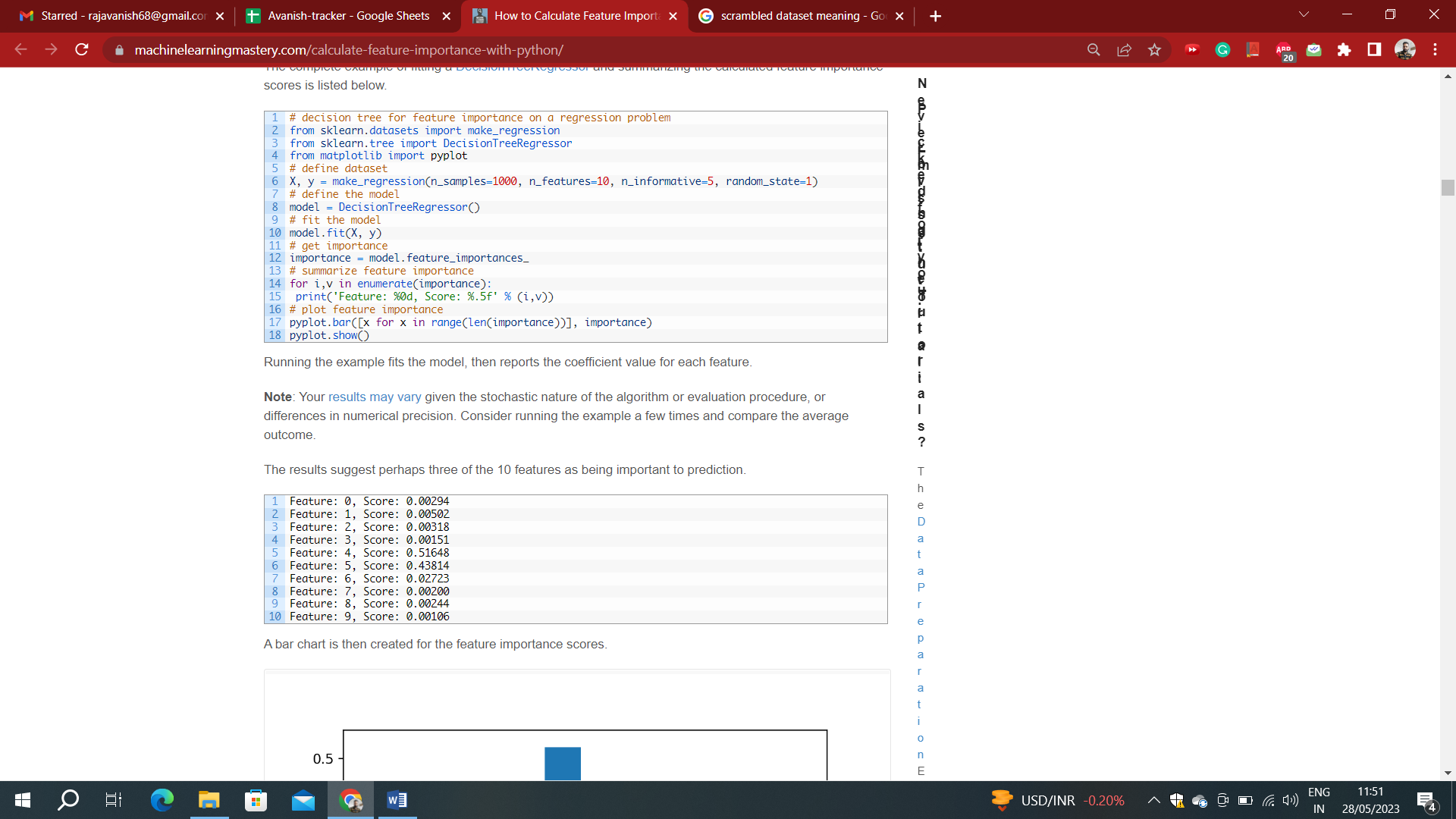
We can use the CART algorithm for feature importance implemented in scikit-learn as the DecisionTreeRegressor and DecisionTreeClassifier classes.

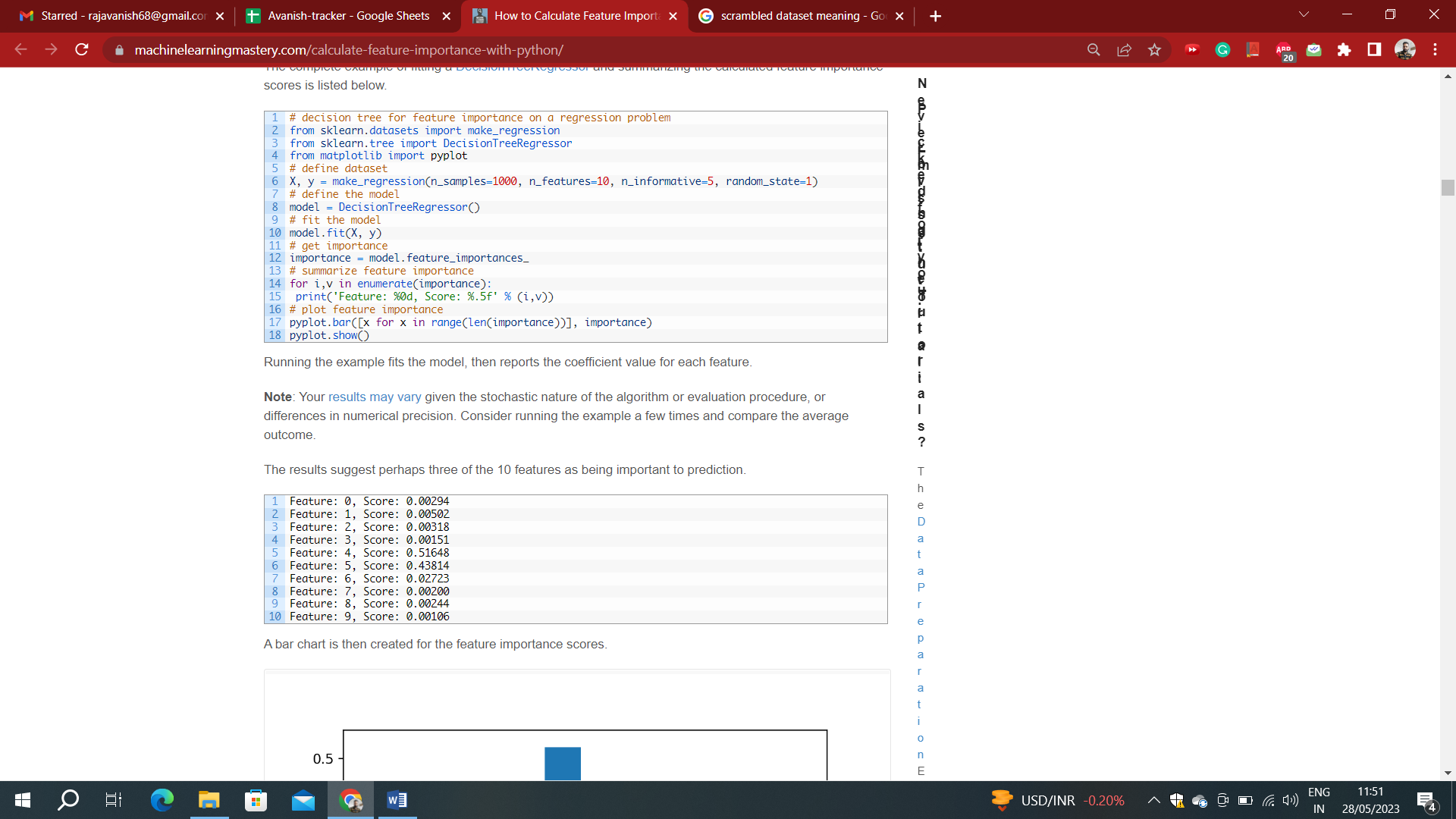
After being fit, the model provides a feature\_importances\_ property that can be accessed to retrieve the relative importance scores for each input feature.

Let’s take a look at an example of this for regression and classification.

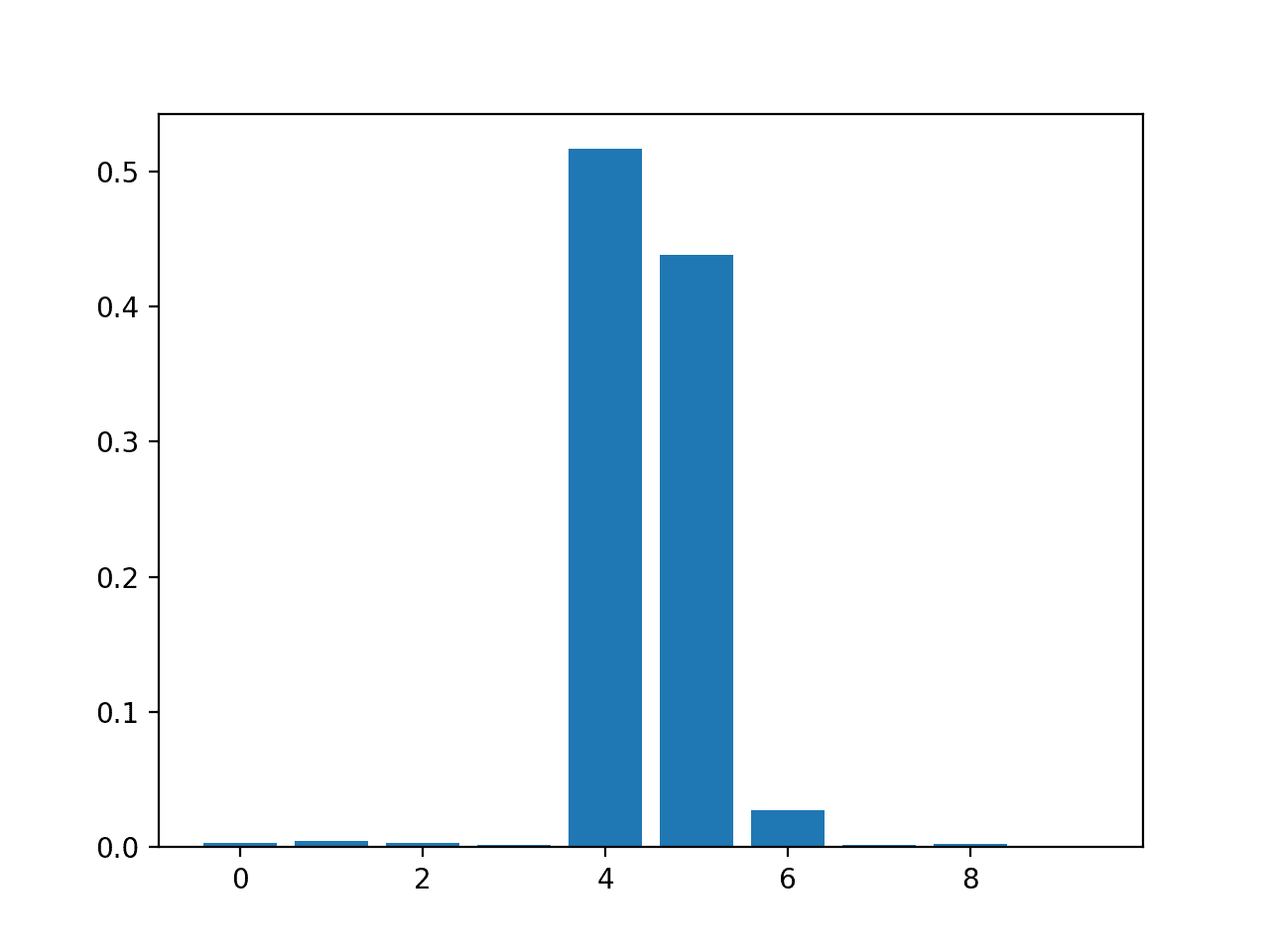
#### **CART Regression Feature Importance**

The complete example of fitting a [DecisionTreeRegressor](https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html) and summarizing the calculated feature importance scores is listed below.





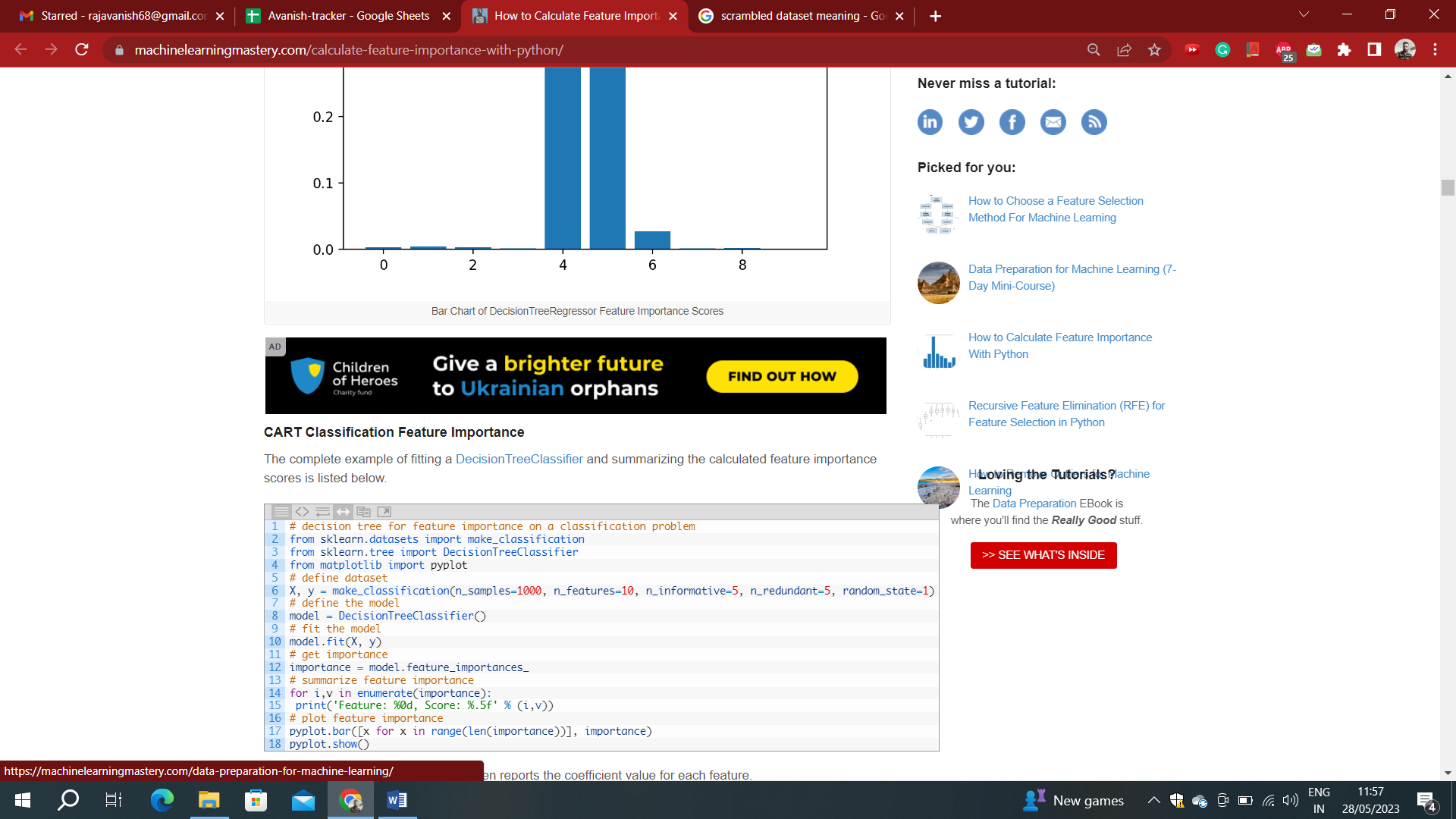
The results suggest perhaps three of the 10 features as being important to prediction.

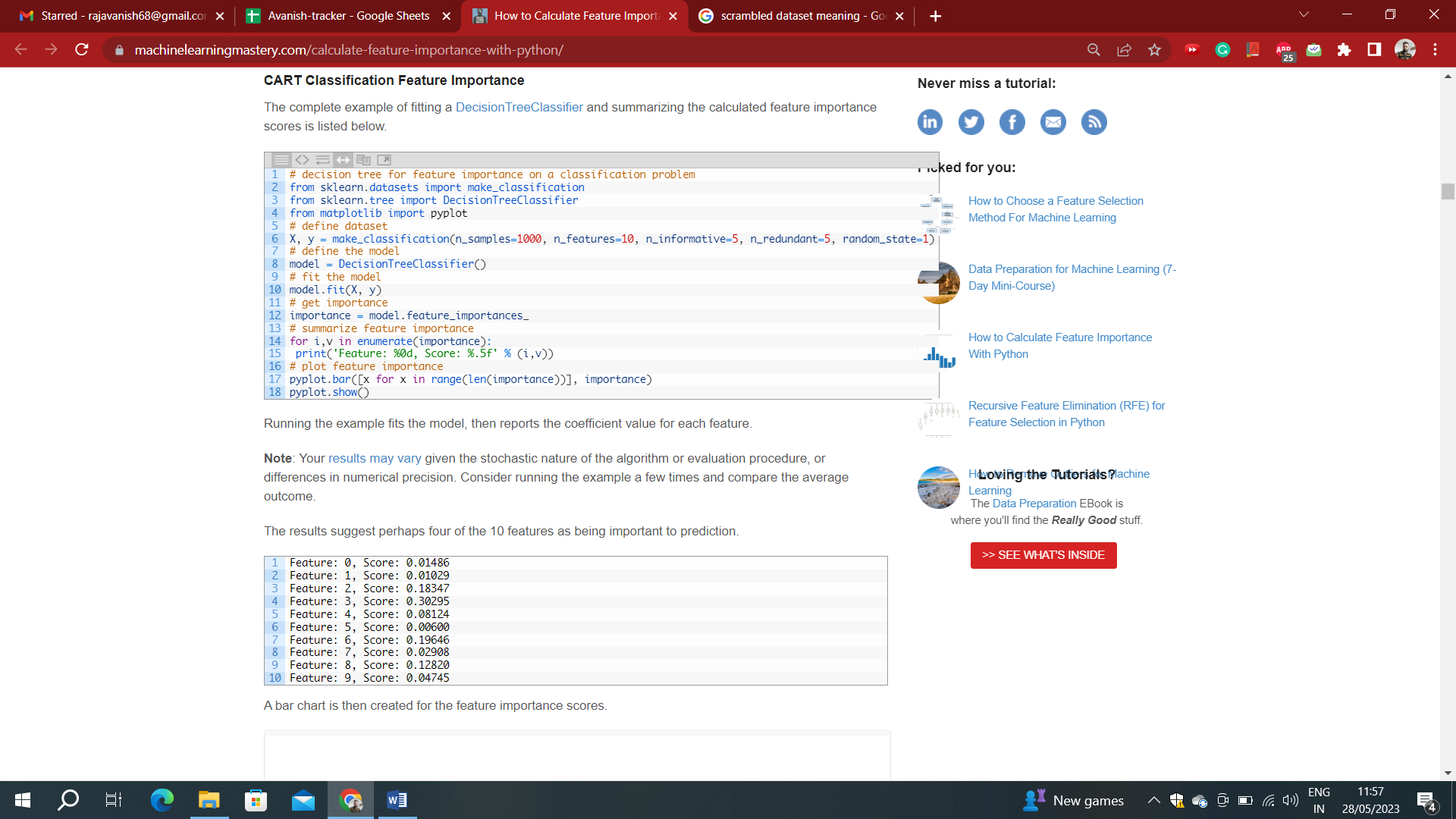


Bar Chart of DecisionTreeRegressor Feature Importance Scores

#### **CART Classification Feature Importance**

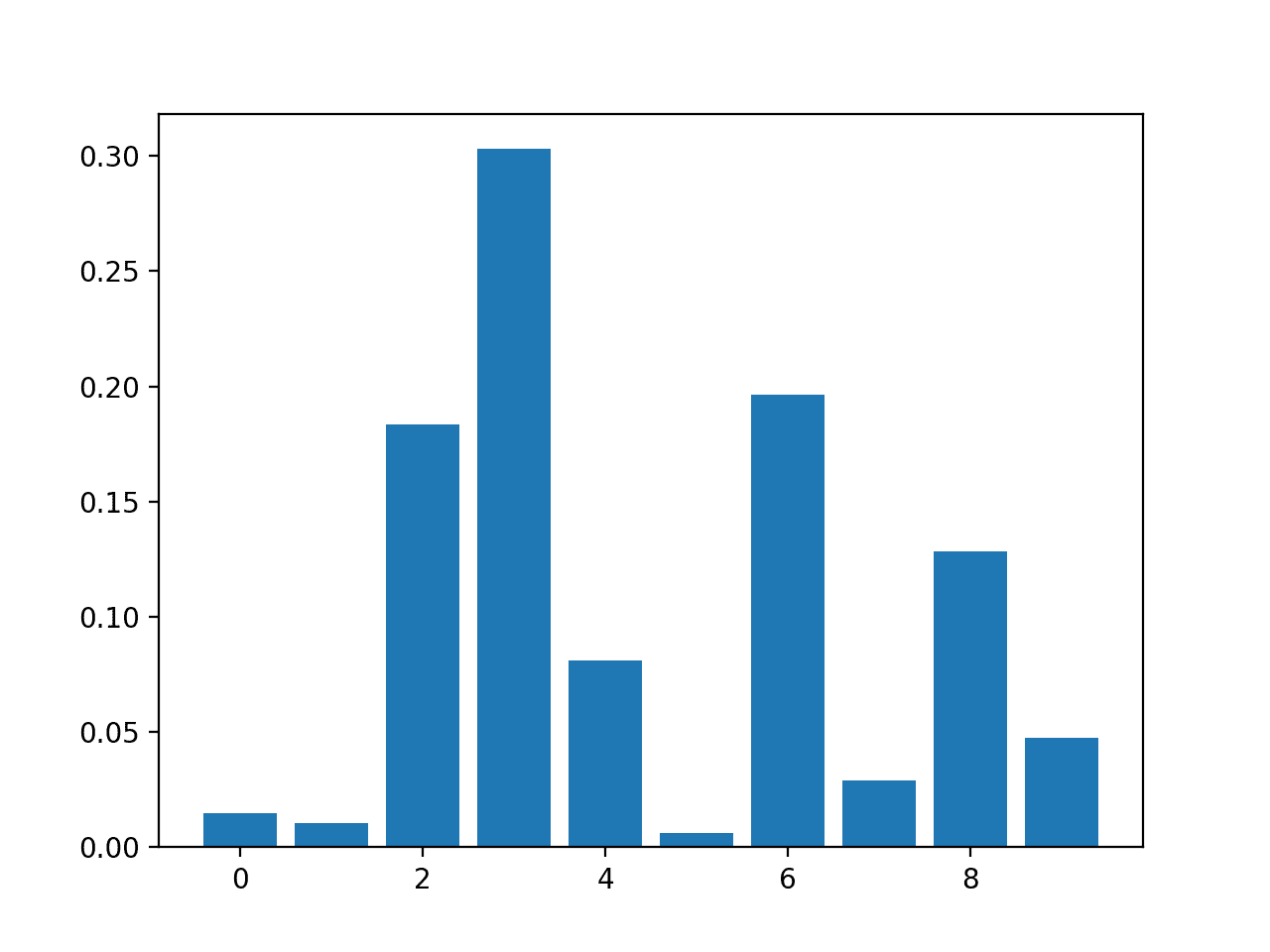
The complete example of fitting a [DecisionTreeClassifier](https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html) and summarizing the calculated feature importance scores is listed below.





The results suggest perhaps four of the 10 features as being important to prediction.

A bar chart is then created for the feature importance scores.



Bar Chart of DecisionTreeClassifier Feature Importance Scores

## **Permutation Feature Importance**

[Permutation feature importance](https://scikit-learn.org/stable/modules/permutation_importance.html) is a technique for calculating relative importance scores that is independent of the model used.

First, a model is fit on the dataset, such as a model that does not support native feature importance scores. Then the model is used to make predictions on a dataset, although the values of a feature (column) in the dataset are scrambled. This is repeated for each feature in the dataset. Then this whole process is repeated 3, 5, 10 or more times. The result is a mean importance score for each input feature (and distribution of scores given the repeats).

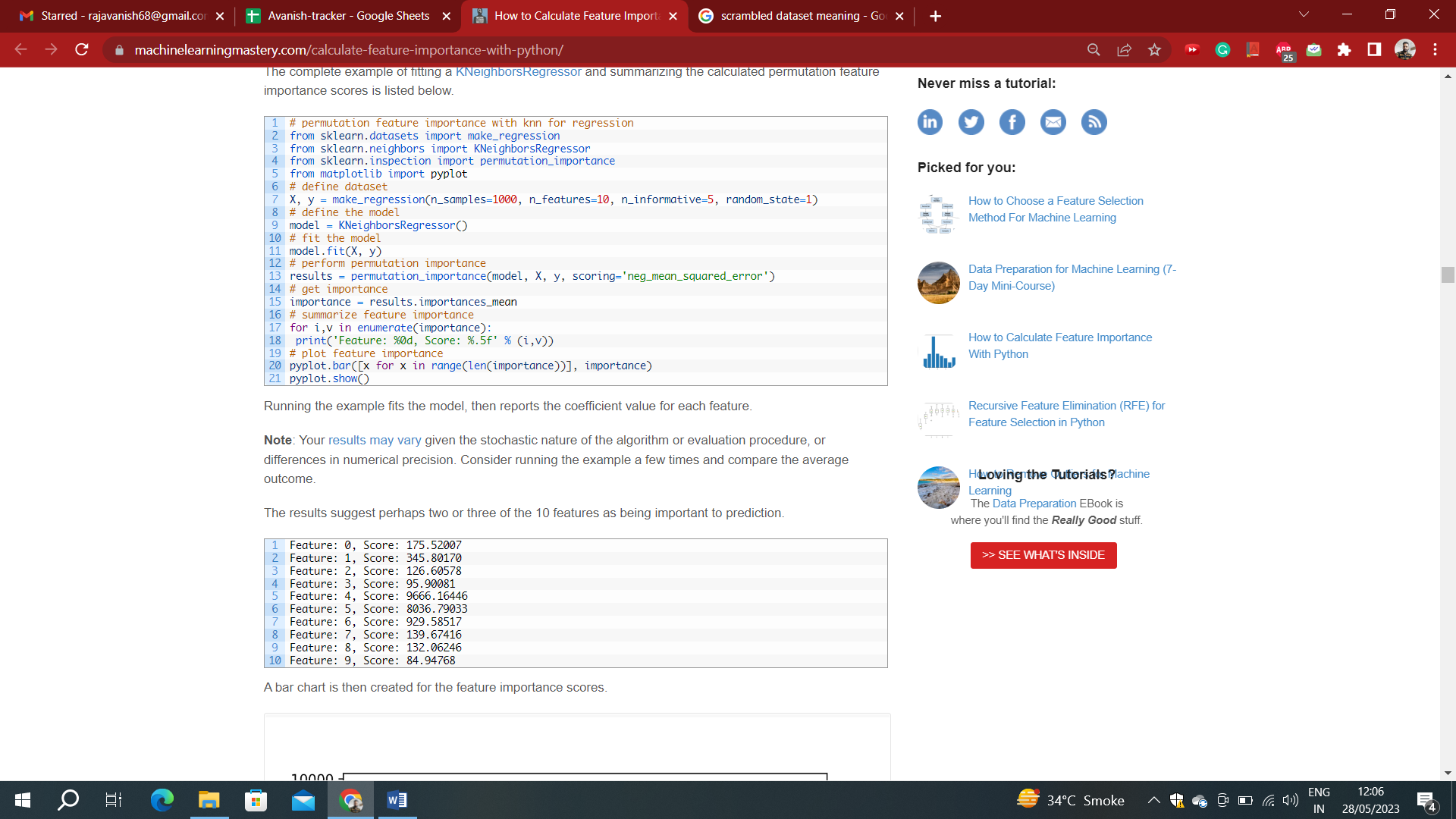
This approach can be used for regression or classification and requires that a performance metric be chosen as the basis of the importance score, such as the mean squared error for regression and accuracy for classification.

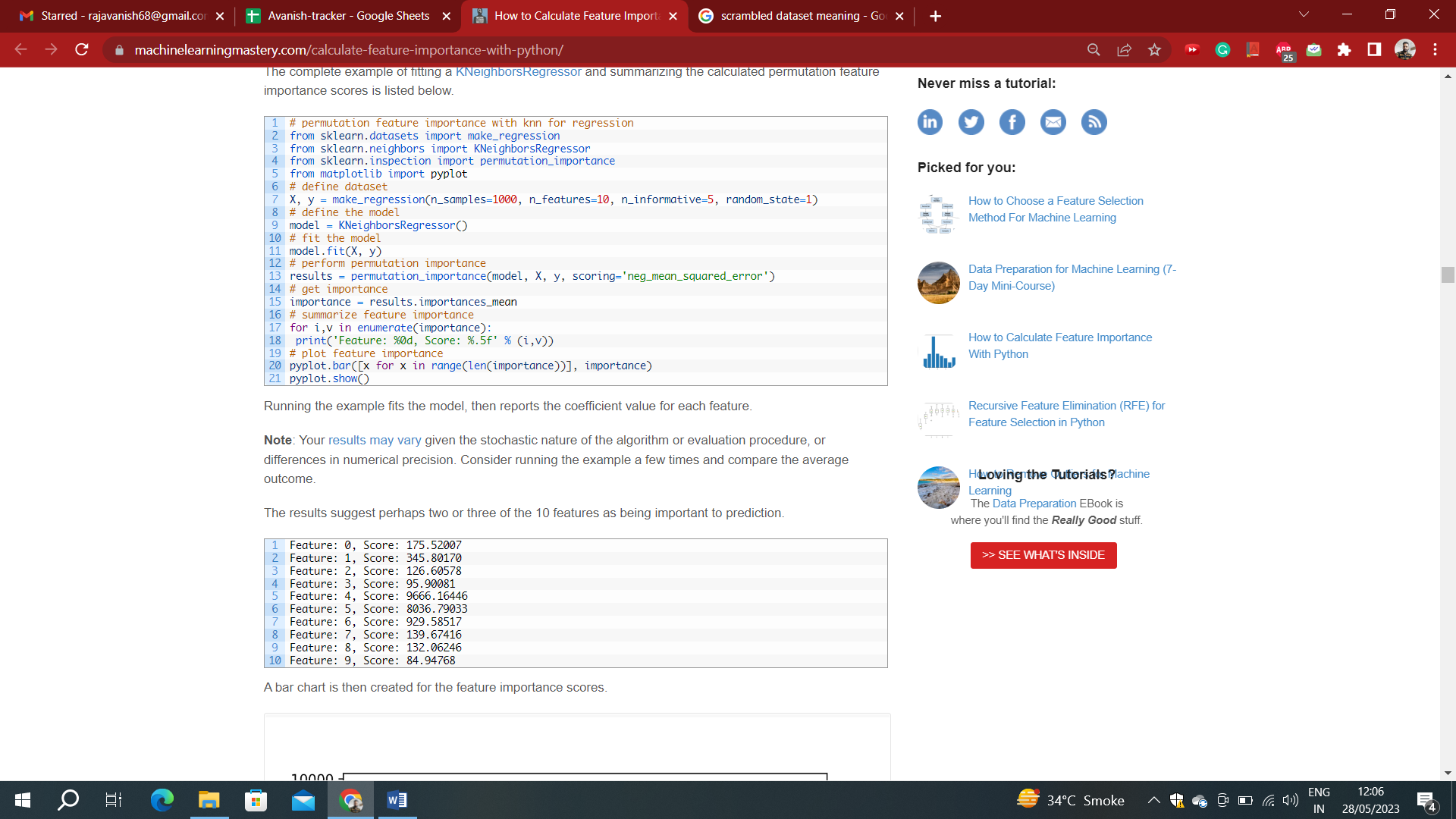
Permutation feature selection can be used via the [permutation\_importance() function](https://scikit-learn.org/stable/modules/generated/sklearn.inspection.permutation_importance.html) that takes a fit model, a dataset (train or test dataset is fine), and a scoring function.

Let’s take a look at this approach to feature selection with an algorithm that does not support feature selection natively, specifically [k-nearest neighbors](https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/).

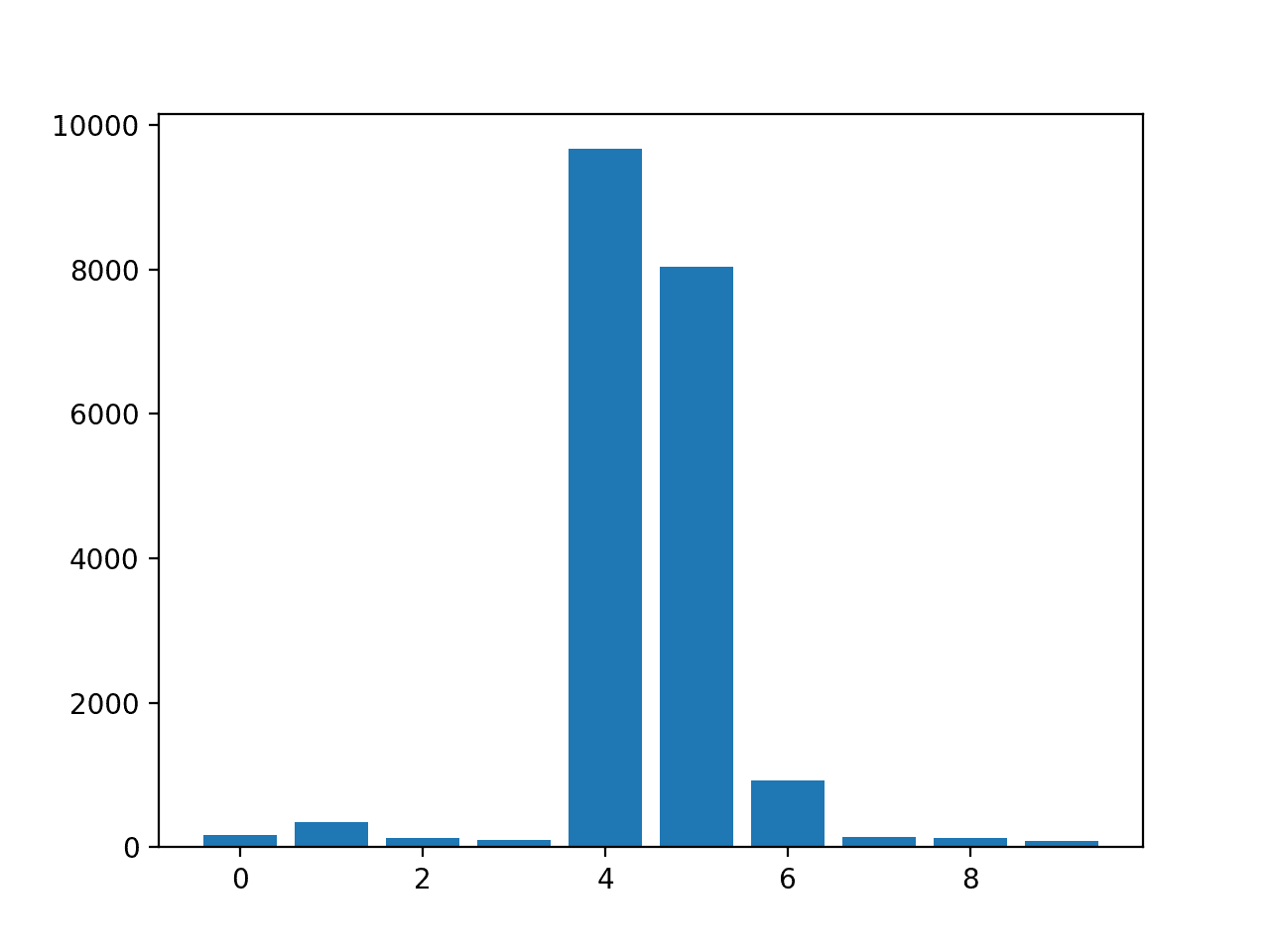
### **Permutation Feature Importance for Regression**

The complete example of fitting a [KNeighborsRegressor](https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html) and summarizing the calculated permutation feature importance scores is listed below.





The results suggest perhaps two or three of the 10 features as being important to prediction.



Bar Chart of KNeighborsRegressor With Permutation Feature Importance Scores

## Note: Similarly, we can do for classification.

## **Feature Selection with Importance**

Feature importance scores can be used to help interpret the data, but they can also be used directly to help rank and select features that are most useful to a predictive model.

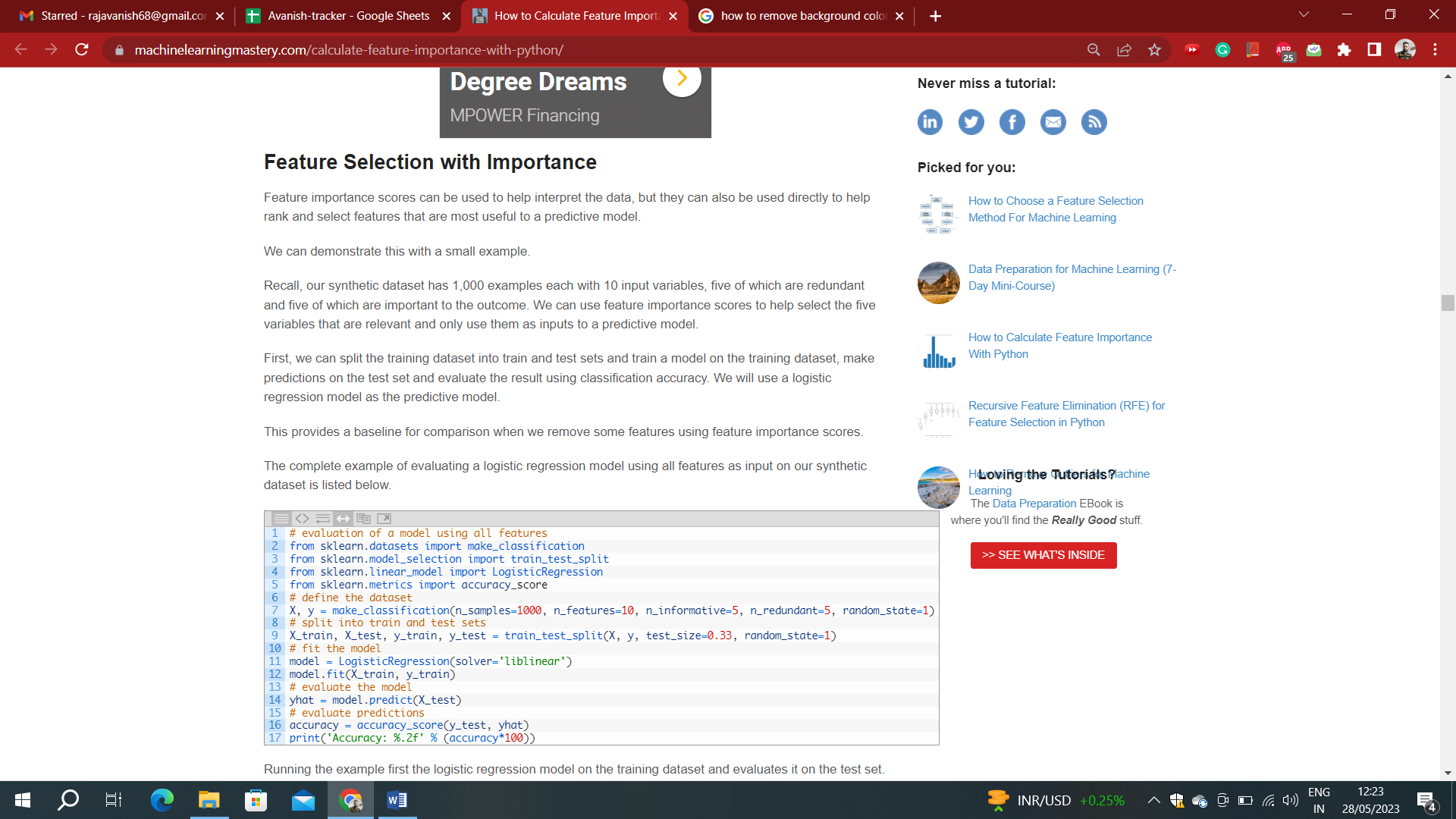
We can demonstrate this with a small example.

Recall, our synthetic dataset has 1,000 examples each with 10 input variables, five of which are redundant and five of which are important to the outcome. We can use feature importance scores to help select the five variables that are relevant and only use them as inputs to a predictive model.

First, we can split the training dataset into train and test sets and train a model on the training dataset, make predictions on the test set and evaluate the result using classification accuracy. We will use a logistic regression model as the predictive model.

This provides a baseline for comparison when we remove some features using feature importance scores.

The complete example of evaluating a logistic regression model using all features as input on our synthetic dataset is listed below.



Accuracy: 84.55

In this case we can see that the model achieved the classification accuracy of about 84.55 percent using all features in the dataset.

Given that we created the dataset, we would expect better or the same results with half the number of input variables.

We could use any of the feature importance scores explored above, but in this case we will use the feature importance scores provided by random forest.

We can use the [SelectFromModel](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectFromModel.html) class to define both the model we wish to calculate importance scores, [RandomForestClassifier](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html) in this case, and the number of features to select, 5 in this case.

# configure to select a subset of features

fs = SelectFromModel(RandomForestClassifier(n\_estimators=200), max\_features=5)

We can fit the feature selection method on the training dataset.

This will calculate the importance scores that can be used to rank all input features. We can then apply the method as a transform to select a subset of 5 most important features from the dataset. This transform will be applied to the training dataset and the test set.

# learn relationship from training data

fs.fit(X\_train, y\_train)

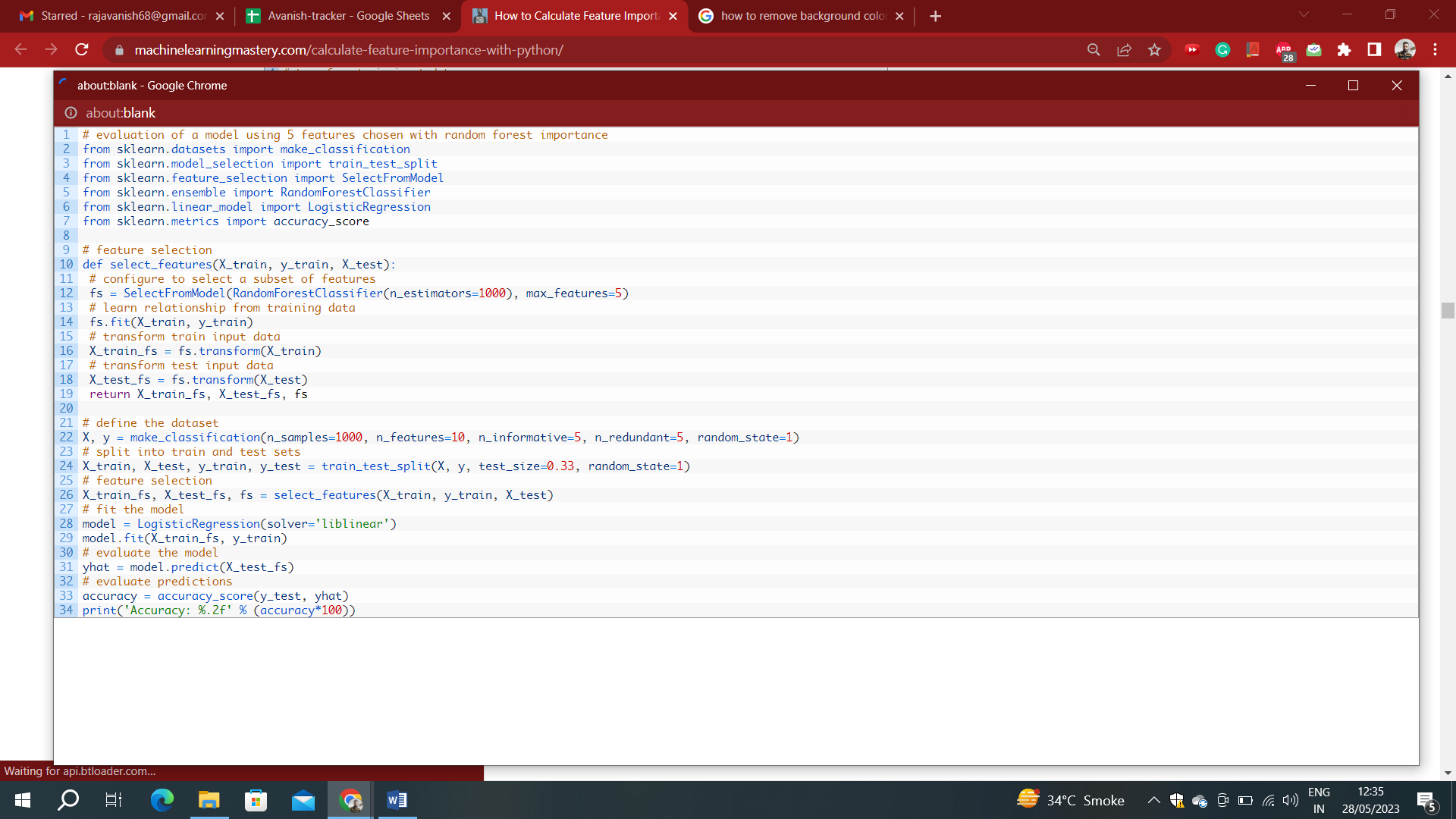
# transform train input data

X\_train\_fs = fs.transform(X\_train)

# transform test input data

X\_test\_fs = fs.transform(X\_test)

Tying this all together, the complete example of using random forest feature importance for feature selection is listed below.



In this case, we can see that the model achieves the same performance on the dataset, although with half the number of input features. As expected, the feature importance scores calculated by random forest allowed us to accurately rank the input features and delete those that were not relevant to the target variable.

Accuracy: 84.55