

Transiting Planet Search in the Kepler Pipeline Using Automated Machine Learning

MSc Research Project
Data Analytics

Martin Mohan
Student ID: X18191339

School of Computing
National College of Ireland

Supervisor: Catherine Mulwa



National College of Ireland
Project Submission Sheet
School of Computing

National
College of
Ireland

Student Name:	Martin Mohan
Student ID:	X18191339
Programme:	Data Analytics
Year:	2020
Module:	MSc Research Project
Supervisor:	Catherine Mulwa
Submission Due Date:	19/08/2020
Project Title:	Transiting Planet Search in the Kepler Pipeline Using Automated Machine Learning
Word Count:	13625
Page Count:	69

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	28th September 2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Transiting Planet Search in the Kepler Pipeline Using Automated Machine Learning

Martin Mohan
X18191339

1 Introduction

This document (Configuration Manual) accompanies the Technical Report of the same title "Transiting Planet Search in the Kepler Pipeline Using Automated Machine Learning". It explains the technical details required to reproduce the hardware/software. Initially the hardware setup is described 2.1. This is followed by a description of the software environment 2 and the software used 3.

An overview of the software pipeline is reproduced in section 4 and the components of the pipeline then discussed individually. The pipeline follows the **SEMMA** approach and this configuration manual is also constructed in this manner. The chapter Sample 5 describes how data was obtained. This is then followed by Explore 6, Modify 7, Model 8 and Assess 9.

The pipeline combined **Transit Crossing Events (TCE)** and **Kepler Objects of Interest (KOI)** to find **Planetary Candidates (PCs)** using machine learning.

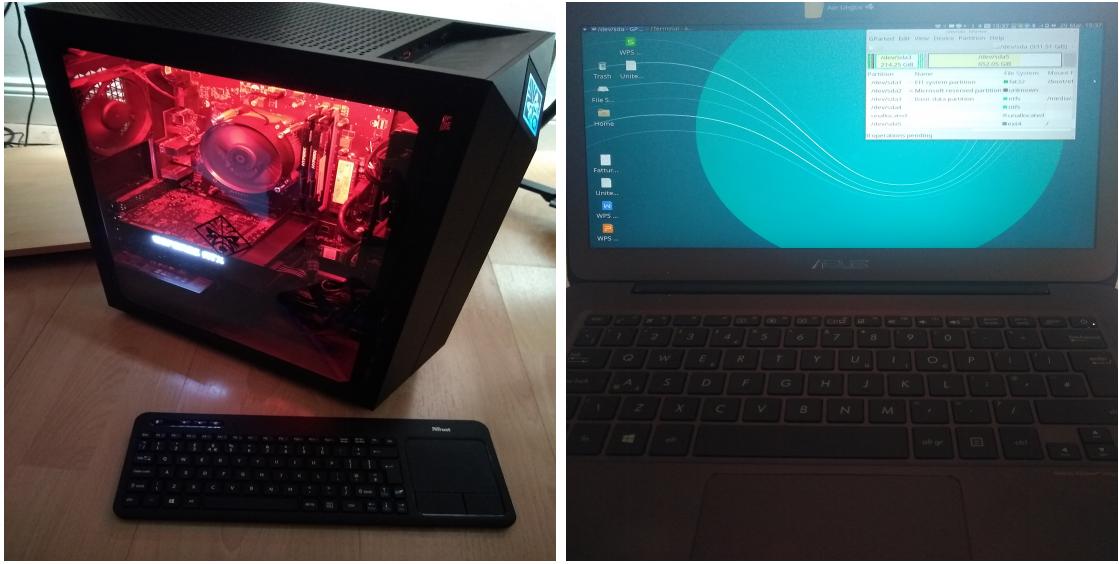
2 Environment Setup

2.1 Computer Hardware

A desktop and laptop running on linux were used for development see Figure 1. The laptop contained 100GB ssd memory and the desktop contained a mixture of ssd and rotating disks.

2.2 Operating System Environment

Both computers in Figure 1 came with Windows 10.0 pre-installed and in both cases the file systems were modified to be dual boot with ubuntu 18.04 and 19.10. Development was performed on the ubuntu environment illustrated in Figure 2 in order to take advantage of the linux command line and shell scripts.



(a) HP - OMEN 875-0051 Gaming PC
2 TB HDD / 256 GB SSD disk
Intel® CoreTM i7, RTX 2060

(b) Asus Zenbook UX30FA laptop
1TB SSD disk
Intel® 5Y10 CPU @ 0.80GHz

Figure 1: Hardware used

3 Software Development Environment

3.1 Package Management with Anaconda

Ubuntu comes with a package manager pre-installed called aptitude but Anaconda¹ was preferred over aptitude because ...

- Virtual environments can be created with Anaconda
- Anaconda packages provide a larger selection and more recent packages.
- Root access is not necessary to install packages

All software packages mentioned were installed using anaconda e.g. To add the caret library to r type.
conda install r-caret - see Figure 3

3.2 Packages Installed with Anaconda

Multiple software was installed on top of the conda environment Figure 3. The list of important software installed for this experiment is shown in Table 1.

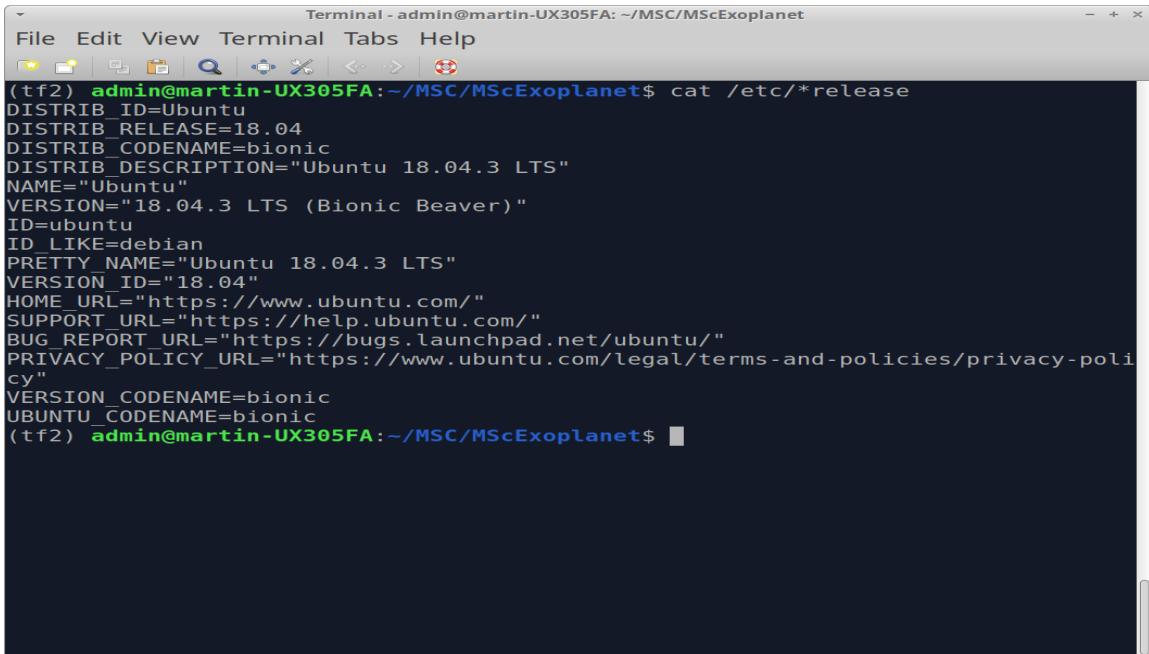
The software installed was used for Treatment ^a options described in the pipeline shown in Figure 6. At each stage of the pipeline csv files were used to store the input and output were possible.

¹<https://phoenixnap.com/kb/how-to-install-anaconda-ubuntu-18-04-or-20-04>



```
Terminal - admin@omen875: ~
File Edit View Terminal Tabs Help
(tf2) admin@omen875:~$ cat /etc/*release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=19.10
DISTRIB_CODENAME=eoan
DISTRIB_DESCRIPTION="Ubuntu 19.10"
NAME="Ubuntu"
VERSION="19.10 (Eoan Ermine)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 19.10"
VERSION_ID="19.10"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=eoan
UBUNTU_CODENAME=eoan
(tf2) admin@omen875:~$
```

(a) Omen desktop ubuntu config



```
Terminal - admin@martin-UX305FA: ~/MSC/MScExoplanet
File Edit View Terminal Tabs Help
(tf2) admin@martin-UX305FA:~/MSC/MScExoplanet$ cat /etc/*release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=18.04
DISTRIB_CODENAME=bionic
DISTRIB_DESCRIPTION="Ubuntu 18.04.3 LTS"
NAME="Ubuntu"
VERSION="18.04.3 LTS (Bionic Beaver)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 18.04.3 LTS"
VERSION_ID="18.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=bionic
UBUNTU_CODENAME=bionic
(tf2) admin@martin-UX305FA:~/MSC/MScExoplanet$
```

(b) Asus ubuntu config

Figure 2: Ubuntu linux environment on Omen Desktop and Asus laptop

The terminal window shows the output of the command `conda list`. The list includes various software packages and their versions, along with their source locations (conda-forge or anaconda). Key packages listed include krb5, lazy-object-proxy, ld_impl_linux-64, libblas, libblas, libcurl, libedit, libffi, libgcc-ng, libgfortran-ng, libgomp, libiconv, liblapack, libopenblas, libpng, libprotobuf, libsodium, libspatialindex, libssh2, libstdcxx-ng, libtiff, libuuid, libwebp-base, libxcb, libxml2, lz4-c, make, markdown, markupsafe, tensorflow, tensorflow-base, tensorflow-estimator, tensorflow-gpu, and tflite-metacompiler.

```

File Edit View Terminal Tabs Help
Terminal - admin@martin-UX305FA: ~
krb5          1.16.4      h2fd8d38_0    conda-forge
lazy-object-proxy 1.4.3       py37hb6447c_0  anaconda
ld_impl_linux-64 2.33.1      h53a641e_7    conda-forge
libblas         3.8.0       14_openblas   conda-forge
libblas         3.8.0       14_openblas   conda-forge
libcurl         7.68.0      hda55be3_0    conda-forge
libedit         3.1.20170329 hf8c457e_1001  conda-forge
libffi          3.2.1       helb5a44_1006  conda-forge
libgcc-ng       9.2.0       h24d8f2e_2    conda-forge
libgfortran-ng 7.3.0       hdf63c60_4    conda-forge
libgomp         9.2.0       h24d8f2e_2    conda-forge
libiconv        1.15        h516909a_1006  conda-forge
liblapack        3.8.0       14_openblas   conda-forge
libopenblas      0.3.7       h5ec1e0e_6    conda-forge
libpng          1.6.37      hbc83047_0   anaconda
libprotobuf     3.11.2      h8b12597_0   conda-forge
libsodium       1.0.17      h516909a_0    conda-forge
libspatialindex 1.9.3       he6710b0_0   anaconda
libssh2         1.8.2       h22169c7_2   conda-forge
libstdcxx-ng    9.2.0       hdf63c60_2    conda-forge
libtiff         4.1.0       hc7e4089_6   conda-forge
libuuid         2.32.1      h14c3975_1000  conda-forge
libwebp-base    1.1.0       h516909a_3    conda-forge
libxcb          1.13        h1bed415_1   anaconda
libxml2         2.9.10      hee79883_0   conda-forge
lz4-c           1.8.3       helb5a44_1001  conda-forge
make            4.3         h516909a_0   conda-forge
markdown        3.1.1       py_0          conda-forge
markupsafe     1.1.1       py37hb6447c_0  anaconda
(tf2) admin@martin-UX305FA: ~$ conda list | grep tensor
tensorboard    2.0.0       pyhb38c66f_1
tensorflow     2.0.0       gbu_py37h768510d_0
tensorflow-base 2.0.0       gbu_py37h0ec5d1f_0
tensorflow-estimator 2.0.0       pyh2649769_0
tensorflow-gpu  2.0.0       h0d30ee6_0
(tf2) admin@martin-UX305FA: ~$
```

Figure 3: Conda Environment

Table 1: Important SW Packages installed

Package	Uses
Rstudio 11	R Interactive statistical analysis and data processing 10
Jupyter Lab	Python3 interactive development and app development.
Argparser lib Python and R	A command line interface to Python and R. Initial development was done in jupyter lab and rstudio but this was then modified to be a command line interface for the pipeline.
IBM SPSS	Used for general statistical analysis.
Linux tools	Shell scripts, Text editing(vim),remote tools rsync, screen, ssh etc ...
texstudio	Texstudio was used for latex documentation and diagrams were designed using the Tikz package.
tpot	Automated software machine learning package Olson and Moore (2019).

3.3 Development Environment

Jupyter lab was used for initial python development see Figure 4 and rstudio was used for initial R development see Figure 5. Once developed the software was put into a command line the see section 4. All other development was performed using the vim text editor see Figure 8 (a).

JupyterLab - Mozilla Firefox

localhost:8888/lab

File Edit View Run Kernel Tabs Settings Help

Name Last Modified

- Treat3.ipynb a month ago
- Treat3.py 6 days ago
- Treat3.sh 5 days ago
- Treat3nn.sh 6 days ago
- Treat4_fit.sh 14 days ago
- Treat4.ipynb** a minute ago
- Treat4.py 4 days ago
- Treat4.res 4 days ago
- Treat4.sh 6 days ago
- Treat4b.res 4 days ago
- Treat4b.sh 4 days ago
- Treat4c.sh 4 days ago
- Treat5.ipynb** a minute ago
- Treat5.py 14 hours ago
- Treat5.py.good 6 days ago
- Treat5.sh 5 days ago
- Treat5b.sh 4 days ago

```

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Run model and generate report files _roc')
    parser.add_argument( "-i", type=str, default="data/TK.csv",
                        help="input file: train_test_split will be run on this file (default: %(default)s)")

    parser.add_argument( "--model", type=str, default="GB",
                        help="Model to test (default: %(default)s)")

    parser.add_argument("--overfit", action="store_true",
                        help="overfit test: Use train data to predict outcome and compare against test")

    parser.add_argument("--fit", action="store_true",
                        help="fit model - otherwise load model from .pickle")

    parser.add_argument("--force", action="store_true",
                        help="force overwrite of results")

    parser.add_argument( "--caption", type=str, default="",
                        help="Caption to add to eventual tex file (must be latex friendly) (default: %(default)s)")

    parser.add_argument("--showmodels", action="store_true",
                        help="Show all models and exit")

```

Saving completed Mode: Command Ln 259, Col 1 Treat4.ipynb

(a) jupyter lab: The argparse package was used to create a command line interface

JupyterLab - Mozilla Firefox

localhost:8888/lab

File Edit View Run Kernel Tabs Settings Help

Name Last Modified

- Treat3.ipynb a month ago
- Treat3.py 6 days ago
- Treat3.sh 5 days ago
- Treat3nn.sh 6 days ago
- Treat4_fit.sh 14 days ago
- Treat4.ipynb** a minute ago
- Treat4.py 4 days ago
- Treat4.res 4 days ago
- Treat4.sh 6 days ago
- Treat4b.res 4 days ago
- Treat4b.sh 4 days ago
- Treat4c.sh 4 days ago
- Treat5.ipynb** a minute ago
- Treat5.py 14 hours ago
- Treat5.py.good 6 days ago
- Treat5.sh 5 days ago
- Treat5b.sh 4 days ago

```

f1=mytreat.cref()
f2=arg.eb

df1=pd.read_csv(f1)
df2=pd.read_csv(f2)

df=df1.merge(df2, indicator=True, how='inner')

```

myfit.py: Loaded GB from data/GB.pickle
Treat5.py:pred: created data/GB/pred.csv
Treat5.py:get_plot: created data/GB/pred_confirm.pdf
Treat5.py:get_confirm: data/GB_confirm.csv
rpt2tex:p1: Created file data/GB_pred_confirm.tex using data/GB_pred_confirm.pdf
Treat5.py:cref: data/GB_confirm.csv+data/LR_confirm.csv -> data/GB_LR_cref.csv
csv2tex: data/GB_LR_cref.csv -> data/GB_LR_cref.tex

CONFIRMED 2262 / 2286 (98.95%) TCE 126 / 24514 (0.51%)

Nr of Planets

CANDIDATE 49 / 1771 (2.77%) FALSE POS 3 / 3963 (0.08%)

Saving completed Mode: Command Ln 1, Col 1 Treat5.ipynb

(b) jupyter lab: matplotlib was used to generate graphics

Figure 4: jupyter lab used for python development

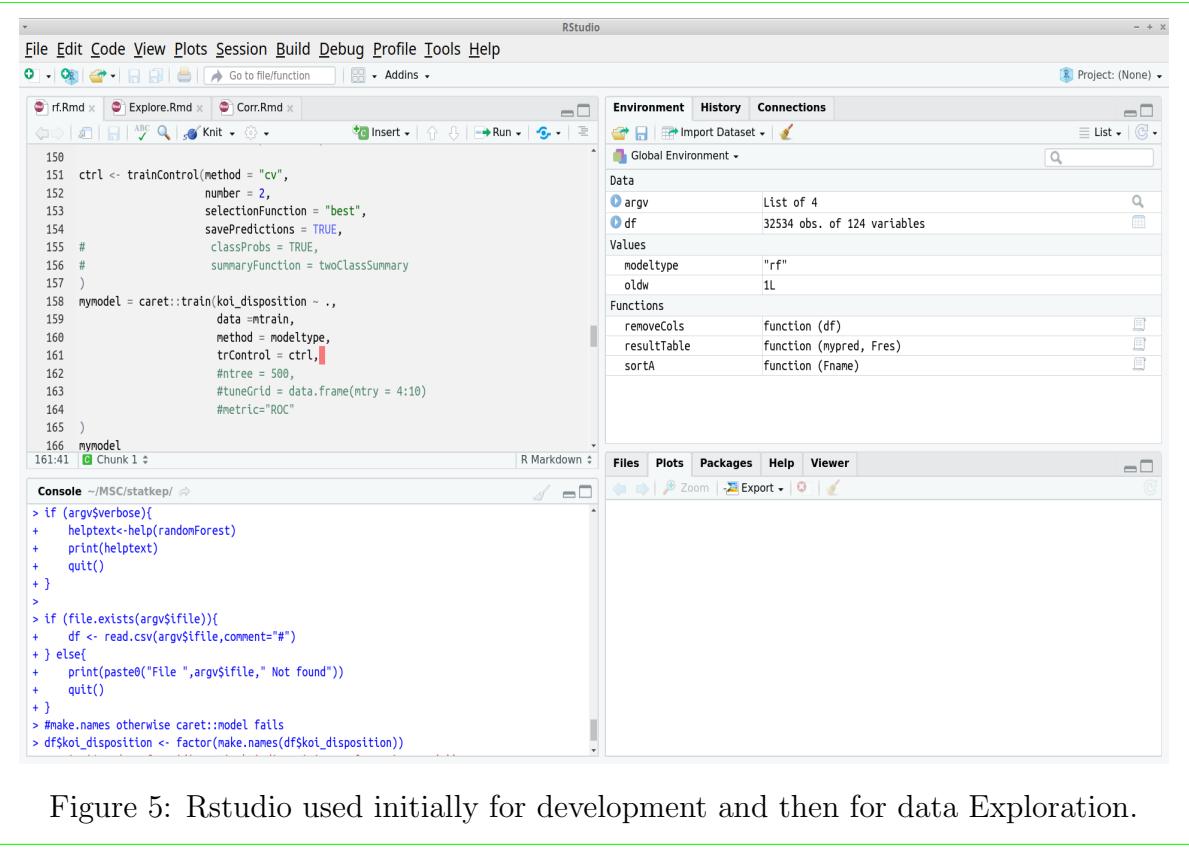
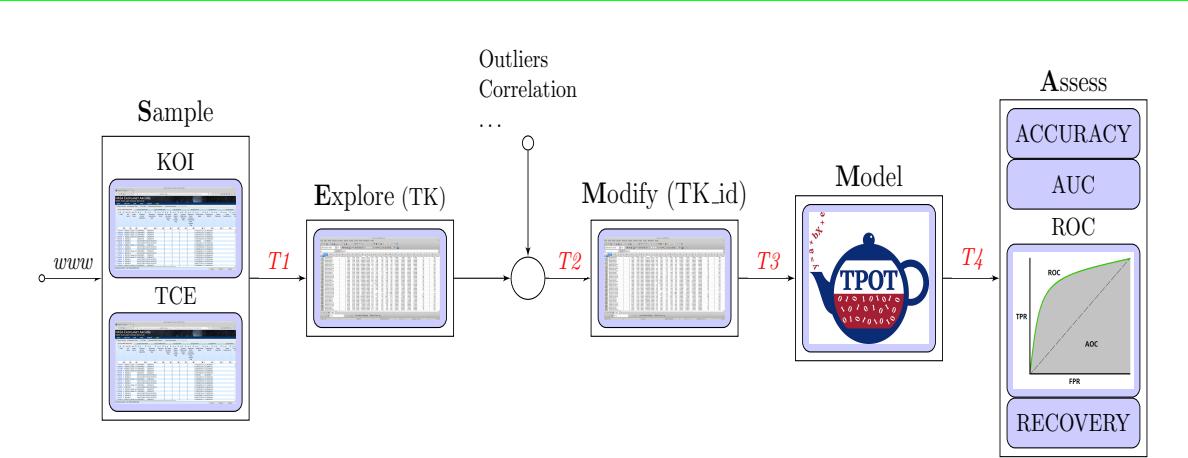


Figure 5: Rstudio used initially for development and then for data Exploration.

4 Pipeline Overview

An overview of the python software pipeline is reproduced in Figure 6 and an overview of the command line are shown in Figure 7.

The programs generated multiple file names. Unit tests were used to test these programs by deleting these files and then checking they are re-created correctly by the programs see Figure 8.



Input	Output	Treat ^a
TCE.csv ² KOI.csv ⁴	TK.csv	T1: TK.csv was a file which was the result of downloading two tables, which were merged and pre-processed to create a single table TK.csv containing multiple IV's and one DV.
TK.csv	TK_id.csv	T2: After exploration TK.csv was modified creating several modifications based on removing correlation or outliers. These modifications were identified using a unique id .
TK_id.csv	TK_id.py	T3: The file TK_id was input to the TPOT software Randal S. Olson and Jason H. Moore (2018) which performs feature selection and tuning in order to provide a python program with the best prediction model.
TK_id.py	Performance results	T4: The models created by TPOT were evaluated against holdout data for Precision, Accuracy and AUC. They were also tested for overfitting.
TK_id.py	Recovery results	T5: The best models were run against the original TCE file to check if any new exoplanets could be found

^a *Treatment* refers to programs which modify or evaluate the data. **T1, T2, T3, T4, T5**

Figure 6: SEMMA pipeline Overview: Each **Treatment** has a defined input and output which is usually a csv file

(a) Treat1.py

```
(base) admin@martin-UX305FA:~/MSC/statkep$ ./Treat1.py -h
usage: Treat1.py [-h] [--tcefile TCEFILE] [--koifile KOIFILE]

1. TCE.csv impute data, drop cols, drop rogue flags 'TCE1.csv' then merge
data/KOI.csv (koi_disposition,kepoi_name) -> 'TK.csv' - This is only run once
at start

optional arguments:
-h, --help      show this help message and exit
--tcefile TCEFILE tce file contains all the IV's (default: data/TCE.csv)
--koifile KOIFILE koi file used to obtain DV (default: data/KOI.csv)
(base) admin@martin-UX305FA:~/MSC/statkep$
```

(b) Treat2.py

```
(base) admin@martin-UX305FA:~/MSC/statkep$ ./Treat2.py -h
usage: Treat2.py [-h] [-iFILE] [-b] [-c1] [-c2] [-pca] [-vif]

Read files data/TK.csv and data/TCE1.csv created by Treat1.py. Create new file
depending on options e.g. -b bTK.csv - Use Treat2.sh to call this

optional arguments:
-h, --help      show this help message and exit
--iFILE         input file (default: data/TK.csv)
--bin           koi_disposition (DV): Create CONFIRMED vs REST file TK=>bTK
(i.e merge CANDIDATE with FALSE POSITIVE n.b this also
creates data/bTCE.csv) (default: False)
--c1            Cap selected outliers (.75pcntile - .25pcntile) which are
not categorical (default: False)
--c2            Cap all outliers (.75pcntile - .25pcntile) which are not
categorical (default: False)
--pca          Product Component Analysis (pc) (default: False)
--vif           Remove multicorrelatd ivs and save _vif (default: False)
(base) admin@martin-UX305FA:~/MSC/statkep$
```

(c) Treat3.py

```
(base) admin@martin-UX305FA:~/MSC/statkep$ ./Treat3.py -h
usage: Treat3.py [-h] [-iFILE] [-mMODEL] [-gGENERATIONS] [-mtmax_time_mins MAX_TIME_MINS]
[-smshowmodels]

Run Tpot model and generate test/train and Tpot file if ifile=data/bT
model=data/bT

optional arguments:
-h, --help      show this help message and exit
--iFILE        Run tpot on ifile create train.csv,test.csv, Tpot.py
(default: data/TK.csv)
--model MODEL  Force Tpot model to choose a model.(Select None or
Tpot light to search for a model) (default: None)
--generations GENERATIONS
Number of generations (default: 100)
--max_time_mins MAX_TIME_MINS
Maximum time to run None for no max: (default: None)
--showmodels   Show all models and exit (default: False)
(base) admin@martin-UX305FA:~/MSC/statkep$
```

(d) Treat4.py

```
(base) admin@martin-UX305FA:~/MSC/statkep$ ./Treat4.py -h
usage: Treat4.py [-h] [-mMODEL] [-f] [-cCAPTION] [-smshowmodels]

Run model and generate report files _roc.pdf, _overfit_roc.pdf, _cm.pdf,
_overfit_cm.pdf, _metric.csv, _tex

optional arguments:
-h, --help      show this help message and exit
--model MODEL  Model to test (also LR) (default: RF)
--fit          fit (and overfit) model - otherwise load model from
.pickle e.g. data/RF.pickle (default: False)
--caption CAPTION Caption to add to eventual tex file (must be latex
friendly) (default: )
--showmodels   Show all models and exit (default: False)
(base) admin@martin-UX305FA:~/MSC/statkep$
```

(e) Treat5.py

```
(base) admin@martin-UX305FA:~/MSC/statkep$ ./Treat5.py -h
usage: Treat5.py [-h] [-f] [-smshowmodels]

Predict confirmed planets using TCE with 4 models GB_vif, RF_vif, LR and DT,
merge into GB_vif RF_vif_LR_DT.csv and then tex files. Then remove entries
from data/EclipsingBinaries

optional arguments:
-h, --help      show this help message and exit
--fit          fit model - otherwise load model from .pickle (default: False)
--showmodels   Show all models and exit (default: False)
(base) admin@martin-UX305FA:~/MSC/statkep$
```

Figure 7: Overview of help files for applications Treat1,2,3,4,5

```

Terminal - statkep_test.sh (~/MSC/statkep) - VIM
File Edit View Terminal Tabs Help
rm data/TK.csv data/TCE1.csv
./Treat1.py > /dev/null
equalF "data/TK.csv" "data/TK.csv.T1"
equalF "data/TCE1.csv" "data/TCE1.csv.T1"
fi

if [ $# -eq 0 ] || [ "${1#*.}" = "T2" ]; then
echo "run ./Treat2.sh test"

rm data/bTCE1_vif_cap2_pca.csv data/TCE1_vif_cap2_pca.csv data/bTK_vif_cap2_pca.csv data/TK_vif_cap2_pca.csv data/bTCE1_vif_cap2.csv d
ata/TCE1_vif_cap2.csv data/bTK_vif_cap2.csv data/TK_vif_cap2.csv data/bTCE1_vif_cap1.csv data/TCE1_vif_cap1.csv data/bTK_vif_cap1.csv
data/TK_vif_cap1.csv data/bTCE1_vif.csv data/TCE1_vif.csv data/bTK_vif.csv data/TK_vif.csv data/bTCE1.csv data/bTK.csv
./Treat2.sh > /dev/null
equalF "data/bTCE1_vif_cap2_pca.csv" "data/bTCE1_vif_cap2_pca.csv.T2"
equalF "data/TCE1_vif_cap2_pca.csv" "data/TCE1_vif_cap2_pca.csv.T2"
equalF "data/bTK_vif_cap2_pca.csv" "data/bTK_vif_cap2_pca.csv.T2"
equalF "data/TK_vif_cap2_pca.csv" "data/TK_vif_cap2_pca.csv.T2"
equalF "data/bTCE1_vif_cap2.csv" "data/bTCE1_vif_cap2.csv.T2"
equalF "data/TCE1_vif_cap2.csv" "data/TCE1_vif_cap2.csv.T2"
equalF "data/bTK_vif_cap2.csv" "data/bTK_vif_cap2.csv.T2"
equalF "data/TK_vif_cap2.csv" "data/TK_vif_cap2.csv.T2"
equalF "data/bTCE1_vif_cap1.csv" "data/bTCE1_vif_cap1.csv.T2"
equalF "data/TCE1_vif_cap1.csv" "data/TCE1_vif_cap1.csv.T2"
equalF "data/bTK_vif_cap1.csv" "data/bTK_vif_cap1.csv.T2"
equalF "data/TK_vif_cap1.csv" "data/TK_vif_cap1.csv.T2"
equalF "data/bTCE1_vif.csv" "data/bTCE1_vif.csv.T2"
equalF "data/TCE1_vif.csv" "data/TCE1_vif.csv.T2"
equalF "data/bTK_vif.csv" "data/bTK_vif.csv.T2"
equalF "data/TK_vif.csv" "data/TK_vif.csv.T2"
equalF "data/bTCE1.csv" "data/bTCE1.csv.T2"
equalF "data/bTK.csv" "data/bTK.csv.T2"

```

(a) Test script bash source code

```

Terminal - admin@martin-UX305FA:~/MSC/statkep
File Edit View Terminal Tabs Help
(base) admin@martin-UX305FA:~/MSC/statkep$ ./statkep_test.sh
./statkep_test.sh T1,T2,T3,T4,T5 or myfit to select test
pass: data/TCE.csv == data/TCE.csv.T0
pass: data/KOI.csv == data/KOI.csv.T0
run ./Treat1.py test
pass: data/TK.csv == data/TK.csv.T1
pass: data/TCE1.csv == data/TCE1.csv.T1
run ./Treat2.sh test
pass: data/bTCE1_vif_cap2_pca.csv == data/bTCE1_vif_cap2_pca.csv.T2
pass: data/TCE1_vif_cap2_pca.csv == data/TCE1_vif_cap2_pca.csv.T2
pass: data/bTK_vif_cap2_pca.csv == data/bTK_vif_cap2_pca.csv.T2
pass: data/TK_vif_cap2_pca.csv == data/TK_vif_cap2_pca.csv.T2
pass: data/bTCE1_vif_cap2.csv == data/bTCE1_vif_cap2.csv.T2
pass: data/TCE1_vif_cap2.csv == data/TCE1_vif_cap2.csv.T2
pass: data/bTK_vif_cap2.csv == data/bTK_vif_cap2.csv.T2
pass: data/TK_vif_cap2.csv == data/TK_vif_cap2.csv.T2
pass: data/bTCE1_vif_cap1.csv == data/bTCE1_vif_cap1.csv.T2
pass: data/TCE1_vif_cap1.csv == data/TCE1_vif_cap1.csv.T2
pass: data/bTK_vif_cap1.csv == data/bTK_vif_cap1.csv.T2
pass: data/TK_vif_cap1.csv == data/TK_vif_cap1.csv.T2
pass: data/bTCE1_vif.csv == data/bTCE1_vif.csv.T2
pass: data/TCE1_vif.csv == data/TCE1_vif.csv.T2
pass: data/bTK_vif.csv == data/bTK_vif.csv.T2
pass: data/TK_vif.csv == data/TK_vif.csv.T2
pass: data/bTCE1.csv == data/bTCE1.csv.T2
pass: data/bTK.csv == data/bTK.csv.T2
run ./Treat3.py test - min of 3 mins
pass: data/TK_100g_None_Tpot.py created
pass: data/TK_100g_Tpot_light_Tpot.py created
pass: data/TK_100g_LR_Tpot.py created
run ./Treat4.py tests
pass: data/Gtest.pickle exists

```

(b) Test script - output

Figure 8: statkep_test.sh: Unit tests for Treat1,2,3,4 and 5. Files removed and unit tests run to see if they are re-created correctly

5 Sample.py Treat1.py

Two files were downloaded and merged from NASA’s exoplanet archive, Threshold crossing event (TCE) ² ³ and KOI.csv ⁴ ⁵. The two files were merged to produce a single csv file called TK.csv with 8020 rows and 126 columns.

The help options of the program used to merge the program are shown in Figure 9 and the treatment involved is summarized in Table 2. Table 3 shows how the tables are merged. Regression tests described in the previous sections were applied to Treat1.py see Figure 8

² table: <https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=tce> last accessed 09/05/2020

³ Description of TCE Data: https://exoplanetarchive.ipac.caltech.edu/docs/API_tce_columns.html last accessed 09/05/2020

⁴ Kepler Object of interest (KOI) Data: <https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=koi> last accessed 09/05/2020

⁵ Description of KOI Data: https://exoplanetarchive.ipac.caltech.edu/docs/API_kepcandidate_columns.html last accessed 09/05/2020

```

Terminal - admin@martin-UX305FA: ~/MSC/statkep
File Edit View Terminal Tabs Help
(base) admin@martin-UX305FA:~/MSC/statkep$ ./Treat1.py -h
usage: Treat1.py [-h] [--tcefile TCEFILE] [--koofile KOIFILE]

1. TCE.csv impute data, drop cols, drop rogue flags 'TCE1.csv' then merge
data/KOI.csv (koi_disposition,kepoi_name) -> 'TK.csv' - This is only run once
at start

optional arguments:
  -h, --help            show this help message and exit
  --tcefile TCEFILE    tce file contains all the IV's (default: data/TCE.csv)
  --koofile KOIFILE    koi file used to obtain DV (default: data/KOI.csv)
(base) admin@martin-UX305FA:~/MSC/statkep$ 
```

Figure 9: Treat1.py help screen

Table 2: Summary of program Treat1.py

Modification	Reason
Merge TCE.csv and KOI.csv	KOI.csv was merged with TCE.csv using the columns kepid, kepoi_name from TCE.csv and kepid and kepoi.name from KOI.csv. koi_disposition is the dependent variable which was saved in 3 levels CANDIDATE, CONFIRMED or FALSE POSITIVE.
Drop	Several columns had the same value for every case. Other columns such as the Autovetter flags (start with av_) were empty and thus dropped
Factorize	Several values were factorised from strings to numbers
Imputation	Missing values were imputed using the 'most frequent' method
Convert signal and noise to .snr	Approximately thirty columns contained signals accompanied by thirty _err columns e.g. tce_period and tce_period_err . An error signal only makes sense unless it is associated with its signal so these were merged to create a signal/noise ratio.

Table 3:

Merge TCE.csv and KOI.csv -> TK.csv

Calculate SNR: **tce_period_sn=tce_period/tce_period_err**

(a) TCE.csv (rows 34032,cols 156)

kepid	tce_plnt_num	tce_period	tce_period_err	etc...
11446443	1	2.47061	0.0000000205109	...
10666592	1	2.20473	0.0000000470213	...
3248033	1	1.33413	0.00000439602	...
7199397	1	105.88	0.000125931	...
10187017	6	22.41	0.000146598	...
...

(b) KOI.csv (rows 8054,cols 138) **koi_disposition** is the **Dependent Variable**

kepid	kepoi_name	koi_disposition	koi_period	etc...
11446443	K00001.01	CONFIRMED	2.470613377	...
10666592	K00002.01	CONFIRMED	2.204735417	...
3248033	K00006.01	FALSE POSITIVE	1.334101467	...
7199397	K00075.01	CANDIDATE	105.8817667	...
10187017	K00082.06	CANDIDATE	22.4098677	...
...

(c) TCE + KOI -> TK.csv

kepid	kepoi_name	koi_disposition	tce_period	tce_period_snr	etc...
11446443	K00001.01	CONFIRMED	2.47061	120453514.960	...
10666592	K00002.01	CONFIRMED	2.20473	46887899.739	...
3248033	K00006.01	FALSE POSITIVE	1.33413	303.49	...
7199397	K00075.01	CANDIDATE	105.88	840.14	...
10187017	K00082.06	CANDIDATE	22.41	152.86	...
...

6 Explore

The program Treat1.py created a single file TK.csv containing 124 IVs and 1 DV(koi_disposition). This was explored using various software like rstudio and spss. Results of the exploration are noted in the subsection below ...

6.1 Missing KOI's

An examination of the difference after merge revealed 25 KOI's were found with no equivalent TCE entry see Table 4

Table 4: Missing KOI's: KOI flags with no equivalent TCE entry

kepid	kepoi_name	koi_disposition	tce_plnt_num
12116489	K00547.03	CANDIDATE	3
10743597	K00989.02	FALSE POSITIVE	2
9411166	K01922.03	CONFIRMED	3
6041734	K02167.03	CONFIRMED	3
11030475	K02248.04	CANDIDATE	4
4247991	K02311.03	CONFIRMED	3
5602588	K02369.03	CANDIDATE	3
11968463	K02433.07	FALSE POSITIVE	7
8753896	K02473.03	CONFIRMED	3
12016267	K02590.02	CANDIDATE	2
9008737	K02768.03	CANDIDATE	3
10793172	K02871.04	CANDIDATE	4
9959492	K03051.02	FALSE POSITIVE	2
4735826	K03184.03	CANDIDATE	3
9008103	K03283.02	CANDIDATE	2
9230021	K03429.04	CANDIDATE	4
3937814	K04084.03	CANDIDATE	3
4548011	K04288.04	CANDIDATE	4
4953173	K04676.02	FALSE POSITIVE	2
8397947	K04772.03	FALSE POSITIVE	3
8973285	K04782.02	CANDIDATE	2
2861126	K04957.02	CANDIDATE	2
6363560	K05270.02	FALSE POSITIVE	2
7935997	K05447.02	CONFIRMED	2
10535708	K05801.02	FALSE POSITIVE	2

6.2 Near Zero Variance

The r-caret function ⁶ nearZeroVar exposed the tce_rogue_flag see Table 5 6.2.1. This flag was removed using Treat1.py but no other changes were made.

Table 5: Near Zero Variance Table: The tce_rogue_flag indicates a case with only 2 transits rather than 3 required for consistency so was removed

-	freqRatio	percentUnique	zeroVar	nzv
tce_rogue_flag	891.111	0.025	0	1
tce_full_conv	25.674	0.025	0	1
tce_sradius_prov	88.122	0.05	0	1
tce_rb_tcount2	19.637	0.71	0	1
tce_rb_tcount3	46.18	0.311	0	1
tce_rb_tcount4	41.129	0.262	0	1

6.2.1 tce_rogue_flag

tce_rogue_flag flags: rogue (or unintended) detections Coughlin et al. (2016) which should not be considered for inclusion in the DR25 KOI activity table ³ but nine cases were detected and removed.

Table 6: rogueFlags 9 rows: According to NASA these should not be in KOI table so they were removed

kepoi_name	kepid	tce_plnt_num	tce_rogue_flag
K04599.01	7335972	1	1
K06152.01	6967278	1	1
K07609.01	10267121	1	1
K07712.01	4996057	1	1
K07880.01	8308516	1	1
K08059.01	11612241	1	1
K08141.01	7686191	1	1
K08176.01	9084832	1	1
K08273.01	8108450	1	1

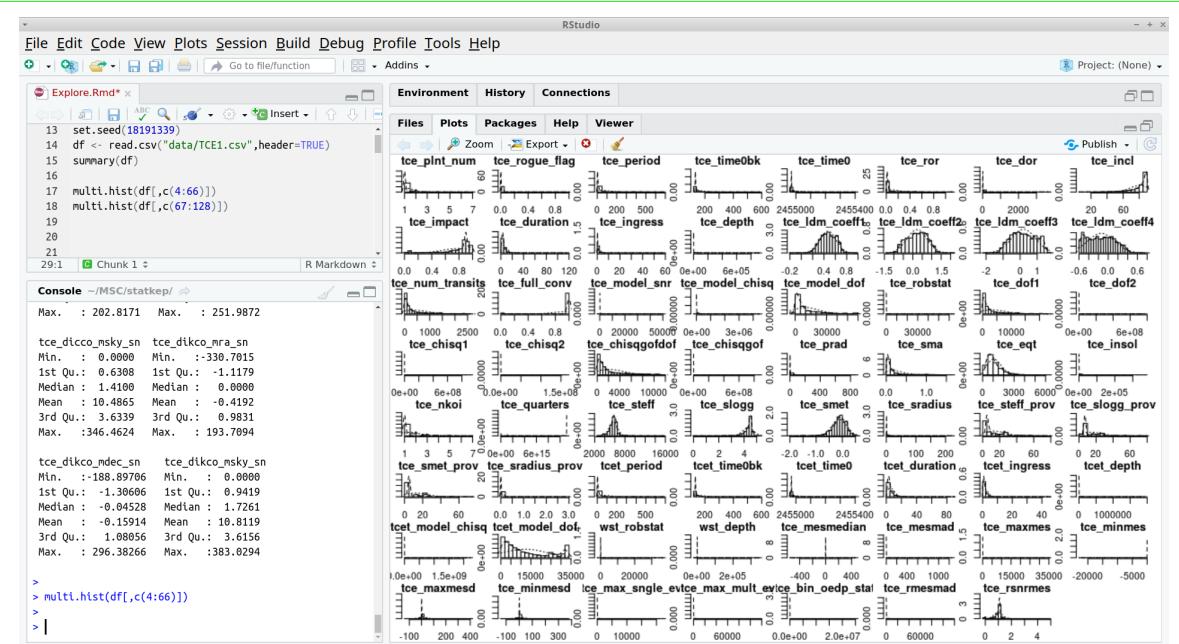
6.3 Outliers

Rstudio and the function multi.hist from the package psych 10 were used to visualize the data. Box plots were viewed in Rstudio see Figure 11.

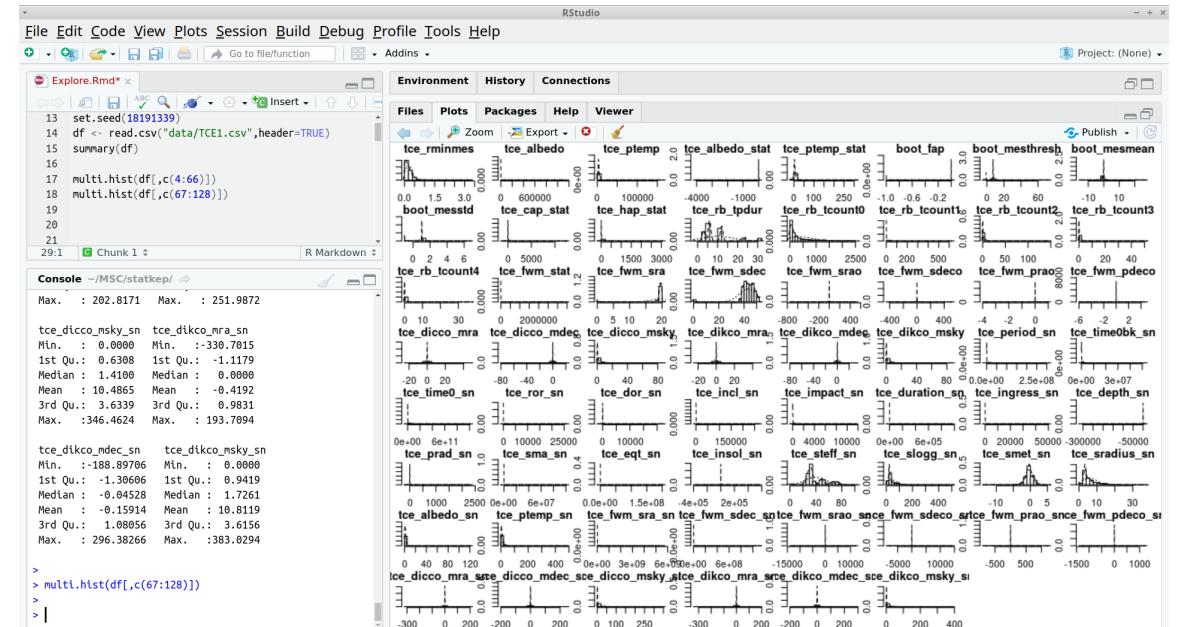
This lead to the removal of tce_quarters seemed to be a large outlier ⁷. Removal of outliers sometimes improved the performance but the improvement was not considered significant for further investigation see the Technical Report.

⁶<https://topepo.github.io/caret/pre-processing.html#zero--and-near-zero-variance-predictors>

⁷The highest outlier tce_quarters actually turned out to be a 12 digit string used for identification rather than a measurement and was removed



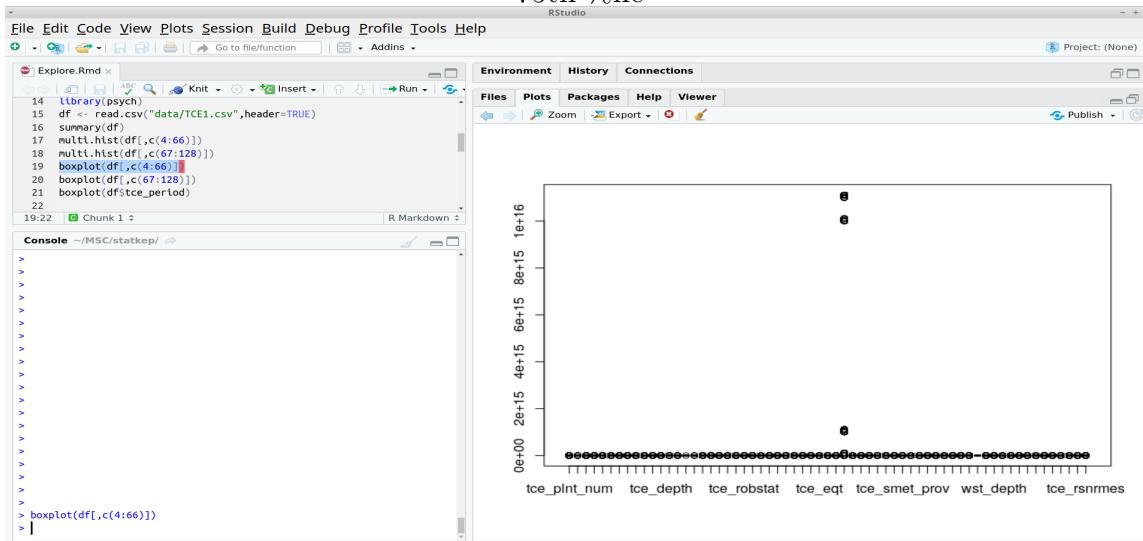
(a) Skew and Kurotosis for results IVs 4:66



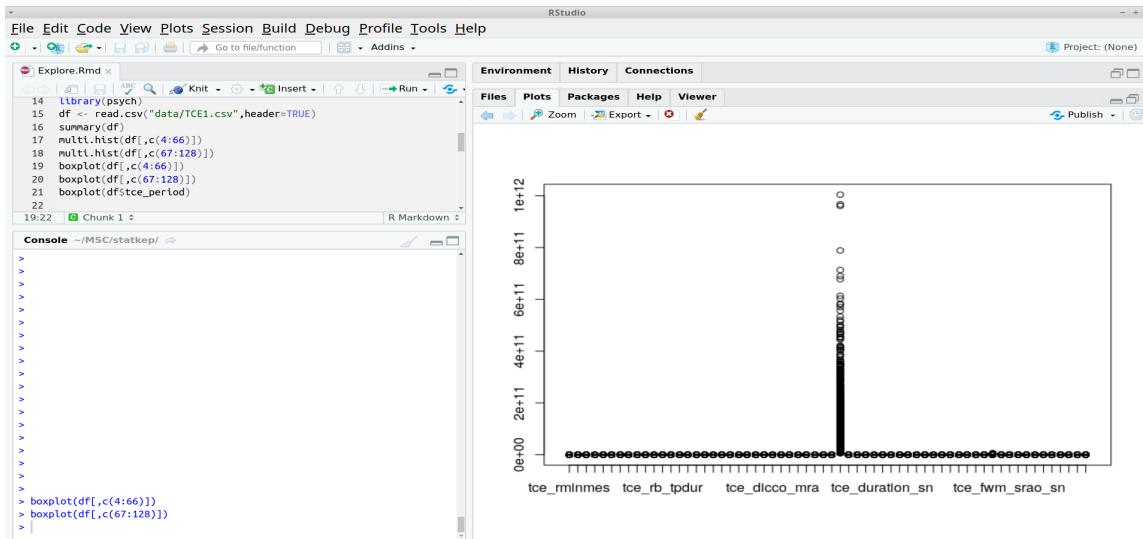
(b) Skew and Kurotosis for IVs 67:128

Figure 10: Graph showing histograms of Independent Variables

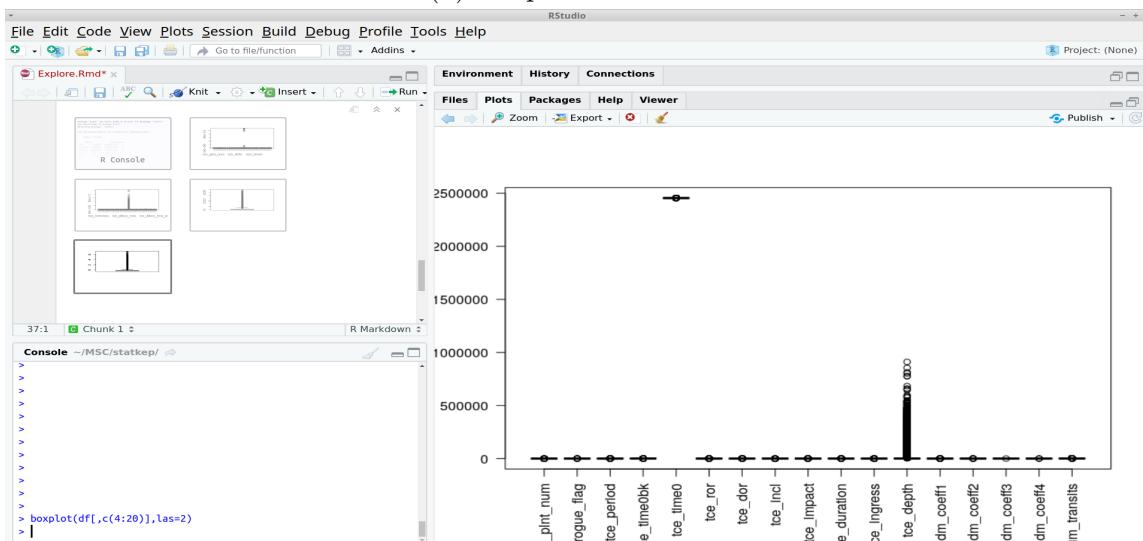
Cap: Interquartile range $3 * IQR$. Replace observations below 25th %ile and above 75th %ile



(a) Boxplots IVs 4:66



(b) Boxplots IVs 67:128



(c) Outliers can be identified by narrowing range

Figure 11: Boxplots of Independent Variables

6.4 Feature Selection

The suitability for factor analysis was assessed using SPSS version 26. Inspection of the correlation matrix revealed the presence of several values of .3 and above. The Kaiser-Meyer-Olkin value was 0.791, exceeding the recommended value of .6 Dziuban and Shirkey (1974) and Bartlett's Test of Sphericity reached statistical significance, supporting the factorability of the correlation matrix.

An inspection of the Total Variance Explained and the Scree plot 12 showed no clean break and 28 components were required to explain 74.2% of the accumulated variance. Reducing the dimensionality of the data set before training a model will usually speed up training, but it may not always lead to a better solution; it all depends on the dataset Géron (2017). In large data sets IVs with small r values may still contribute significantly to prediction results. It was decided not to use the factor analysis as the loss in variance would not justify any gain in speed and **TPOT** performs automatic feature selection during tuning.

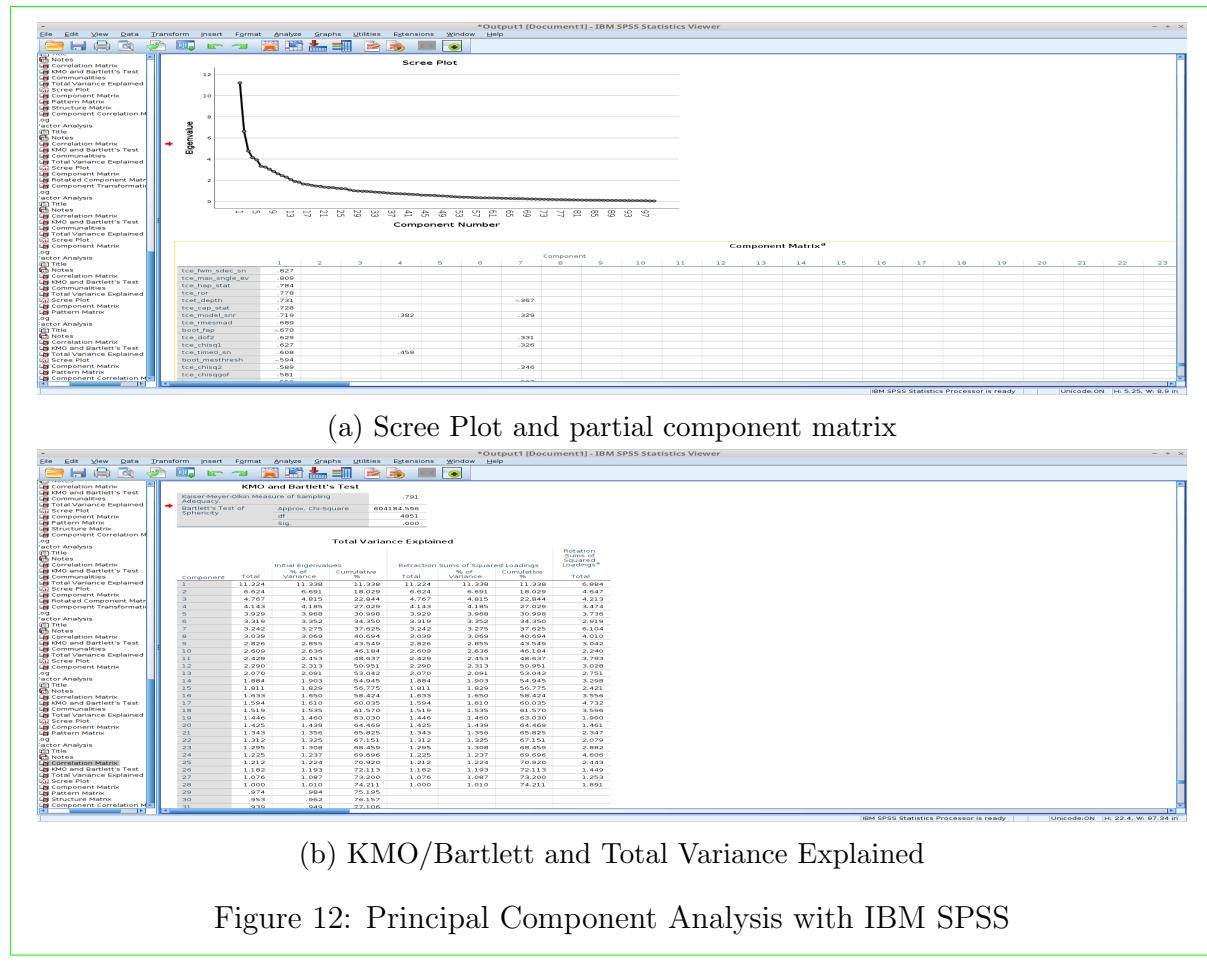


Figure 12: Principal Component Analysis with IBM SPSS

6.5 Correlation with Dependent Variable (koi_disposition)

High correlation of the DV (koi_disposition) with IV's is desirable so a table was created to explore IVs which contributed most to correlation see Table 7.

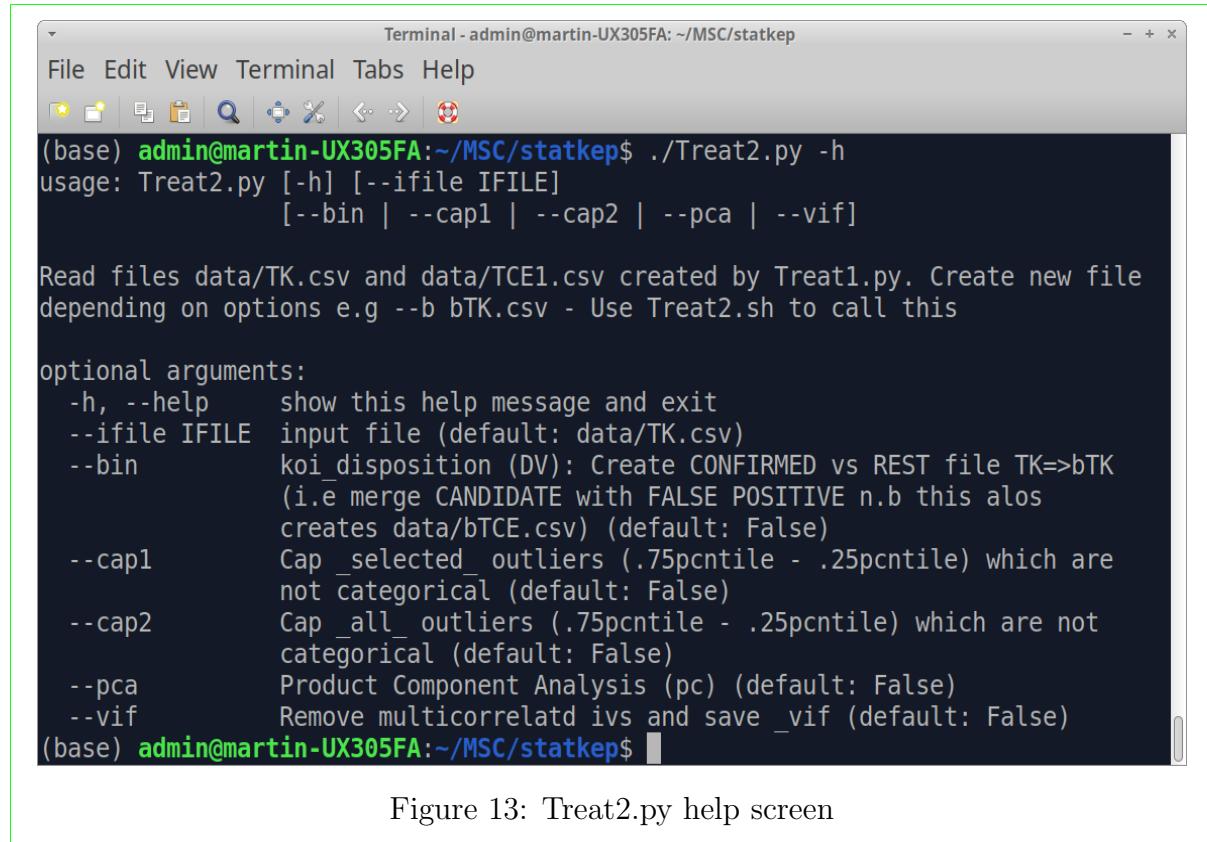
Table 7: koi_disposition vs highest correlated values

Nr	Name	R	Meaning
1	tce_nkoi	-0.3895	Number of Associated KOIs
2	tce_steff_sn	-0.3814	Stellar Effective Temperature [K]
3	tce_ror	0.3810	Planet-Star Radius Ratio
4	tce_dicco_msky	0.3579	PRF $\Delta\theta_{\text{OOT}}$ (arcsec)
5	tce_dikco_msky	0.3556	PRF $\Delta\theta_{\text{KIC}}$ (arcsec)
6	tce_chisqgofdof	0.3537	Chi-Square GOF DOF
7	tce_model_dof	0.3485	Degrees of Freedom
8	tcet_model_dof	0.3335	Degrees of Freedom
9	tce_incl	-0.3236	Inclination [deg]
10	tce_hap_stat	0.3220	Ghost Halo Aperture Statistic
11	tce_sradius_sn	-0.3184	Stellar Radius signal/noise ratio [Solar radii]
12	boot_fap	-0.3132	Bootstrap False Alarm Probability
13	tce_dikco_msky_sn	0.3117	PRF $\Delta\theta_{\text{KIC}}$ sn ratio (arcsec)
14	tce_eqt	0.3104	Equilibrium Temperature [K]
15	tce_dicco_msky_sn	0.3080	PRF $\Delta\theta_{\text{OOT}}$ sn[arcsec]
16	tce_smet_prov	-0.3070	Stellar Metallicity Provenance
17	tcet_ingress	0.3059	Ingress Duration [hrs]
18	tce_rb_tcount0	0.3036	Transit Counts 0
19	tce_steff_prov	-0.3015	Stellar Effective Temperature Provenance
20	tce_dof1	0.3005	Degrees of Freedom 1
21	tce_rsnrmes	-0.3000	Calculated Ratio SNR over MES
22	tce_depth	0.2941	Transit Depth [ppm]
23	tce_num_transits	0.2897	Number of Transits

7 Modify: Treat2.py

Highly correlated IVs are undesirable and were removed. The values to remove were determined by sorting IVs and removing those with the highest correlation iteratively. This was repeated until all those with a VIF > 10 were removed. The values with high VIF were noted and included in the program Treat2.py in order to remove these values automatically.

Treat2.py was used to clean the data. The structure of the program is shown in Figure 14. The help options of the program used to merge the program are shown in Figure 13 and the treatment involved is summarized in Table 8.



The image shows a terminal window titled "Terminal - admin@martin-UX305FA: ~/MSC/statkep". The window contains the help output for the "Treat2.py" script. The output includes the usage command, a note about reading files, optional arguments with their descriptions, and the copyright information at the bottom.

```
File Edit View Terminal Tabs Help
Terminal - admin@martin-UX305FA: ~/MSC/statkep
(base) admin@martin-UX305FA:~/MSC/statkep$ ./Treat2.py -h
usage: Treat2.py [-h] [--ifile IFILE]
                  [--bin | --cap1 | --cap2 | --pca | --vif]

Read files data/TK.csv and data/TCE1.csv created by Treat1.py. Create new file
depending on options e.g --b bTK.csv - Use Treat2.sh to call this

optional arguments:
-h, --help      show this help message and exit
--ifile IFILE  input file (default: data/TK.csv)
--bin          koi_disposition (DV): Create CONFIRMED vs REST file TK=>bTK
              (i.e merge CANDIDATE with FALSE POSITIVE n.b this also
              creates data/bTCE.csv) (default: False)
--cap1         Cap _selected_ outliers (.75pcntile - .25pcntile) which are
              not categorical (default: False)
--cap2         Cap _all_ outliers (.75pcntile - .25pcntile) which are not
              categorical (default: False)
--pca          Product Component Analysis (pc) (default: False)
--vif          Remove multicorrelatd ivs and save _vif (default: False)
(base) admin@martin-UX305FA:~/MSC/statkep$
```

Figure 13: Treat2.py help screen

Table 8: Summary of program Treat2.py. Only highlighted options were used in the final report

Modification	Reason
bin	Investigations were carried out in changing from a multi-class system with 3 levels CANDIDATE, CONFIRMED or FALSE POSITIVE to a system with 2 levels CANDIDATE and FALSE POSITIVE. This was found to improve precision more but more programming is needed. The option is retained as it may be used in future work .
cap1	Capping was done by examining each curve individually and deciding on whether it should be capped but examining each IV individually proved too time consuming to be completed although the option is retained as it may be used in future work .
cap2	All values were capped at $3*(.25 \text{ \%qtile to } .75 \text{ \%qtile})$. If this was applied to an input file the output file was automatically renamed with _cap2. $3*IQR$ was chosen after manual inspection of resulting output graphs using matplotlib. When this option is chosen a new filename with ”_cap2” extension is created.
pca	PCA was attempted using python’s pca software but inconclusive results were obtained so ibm-spss was used to investigate further. The option is retained as it may be useful for other projects in future work .
vif	Multicorrelated values were removed iteratively with the help of python program and jupyter labs. The Variant Inflation factor of each IV was calculated and sorted by size. The largest value was then removed and the process repeated until all values with $VIF > 10$ had been removed. The final list of multicorrelated IVs was included in this program. This option will remove the IVs marked as having high vif from the data file and create a new filename with a ”_vif” extension.

```
Terminal - admin@martin-UX305FA:~/MSC/statkep
File Edit View Terminal Tabs Help
CLASSES
builtins.object
Treat2

class Treat2(builtins.object)
| Treat2(ifile)

    This class will modify the input file depending on the user selection.
    After modification name of the output file is modified to reflect the change.
    e.g. if outlier removal applied ifile.csv -> ifile_out.csv
    Only one method can be called at time.
    The TCE file is modified the same way as the TK file

    Methods defined here:

        __init__(self, ifile)
            The input file is saved

        bin(self)

        cap1(self)

        cap2(self)

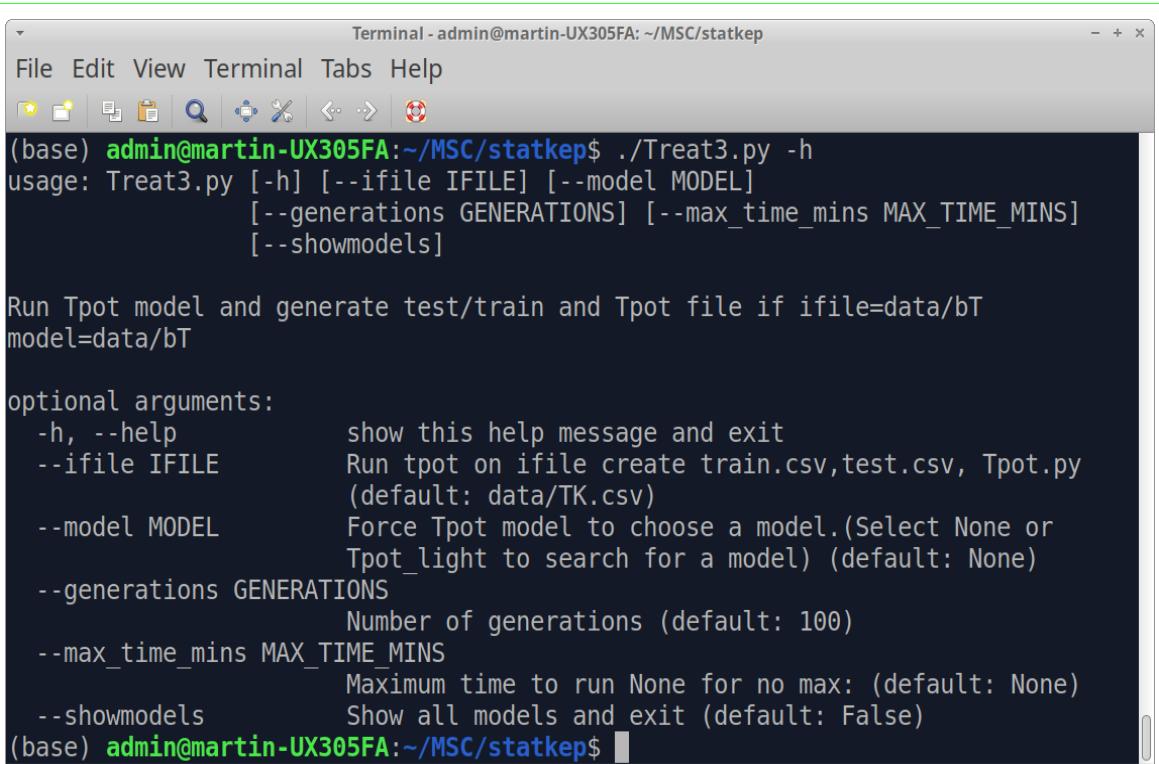
        capOutliers(self, iv)
            Cap all outliers Interquartile range IQR*1.5

        get_vif(self, exogs, data)
            From https://stackoverflow.com/questions/42658379/variance-inflation-factor-in-python
            Return VIF (variance inflation factor) DataFrame
```

Figure 14: Treat2.py methods: To view: import Treat2;help(Treat2)

8 Model Creation using TPOT scripts: Treat3.py

Treat3.py was a wrapper used to create TPOT scripts. The help options of the program used to merge the program are shown in Figure 15 and the treatment involved is summarized in Table 9. After the model is configured it is run on the production desktop and generates a python script with optimized pipeline code see Figure 16. Each generated code is copied as a method to a python class mmodels.py see Figure 17 and this is used to assess the model by the programs Treat4.py and Treat5.py.



The screenshot shows a terminal window titled "Terminal - admin@martin-UX305FA: ~/MSC/statkep". The command "../Treat3.py -h" is entered, displaying the help documentation for the program. The help text includes usage instructions, optional arguments, and detailed descriptions for each argument. The terminal prompt "(base)" appears at the end of the help output.

```
File Edit View Terminal Tabs Help
(base) admin@martin-UX305FA:~/MSC/statkep$ ./Treat3.py -h
usage: Treat3.py [-h] [--ifile IFILE] [--model MODEL]
                  [--generations GENERATIONS] [--max_time_mins MAX_TIME_MINS]
                  [--showmodels]

Run Tpot model and generate test/train and Tpot file if ifile=data/bT
model=data/bT

optional arguments:
  -h, --help            show this help message and exit
  --ifile IFILE        Run tpot on ifile create train.csv,test.csv, Tpot.py
                       (default: data/TK.csv)
  --model MODEL         Force Tpot model to choose a model.(Select None or
                       Tpot_light to search for a model) (default: None)
  --generations GENERATIONS
                        Number of generations (default: 100)
  --max_time_mins MAX_TIME_MINS
                        Maximum time to run None for no max: (default: None)
  --showmodels          Show all models and exit (default: False)
(base) admin@martin-UX305FA:~/MSC/statkep$
```

Figure 15: Treat3.py help screen

Table 9: Summary of program Treat3.py

Modification	Reason
ifile	The file used by TPOT to generate automated machine learning code. The default is TK.csv but if data is modified e.g. multicorrelation it is Tk_vif.csv
model	Restrict models tpots uses e.g.. 'TPOT light' uses lighter models. See --showmodels for models available. Only TPOT light and default TPOT produced significant results.
generations	Default is 100 but you can use less for faster models
showmodels	Show models available

```

File Edit View Terminal Tabs Help
Terminal - Treat3.py (~/MSC/statkep) - VIM
X train=train.drop(['kepid','koi_disposition','kepoi_name'], axis=1).to_numpy()
y_train, uniques =pd.factorize(train['koi_disposition'],sort=False)
X test=test.drop(['kepid','koi_disposition','kepoi_name'], axis=1).to_numpy()
y_test, uniques =pd.factorize(test['koi_disposition'],sort=False)

# Make config dict
if model=="None": config_dict=None
elif model=="light": config_dict=models[model]
else:
    key=models[model]
    value="{{ }}"
    config_dict={key:{}}

sdate=datetime.now().strftime("%d/%m/%Y %H:%M:%S")
Tpot = TPOTClassifier(verbosity=2,generations=generations,config_dict=config_dict,max_time_mins=max_time_mins,random_state=random_state,EARLY_STOP=EARLY_STOP)
Tpot.fit(X_train, y_train)
Tpot_score=Tpot.score(X_test, y_test)
Tpot_file=file.replace(".csv","_Tpot.py")
Tpot.export(Tpot_file)
print(f"Score see {Tpot_file}") # The correct score is in the tpot export file
edate=datetime.now().strftime("%d/%m/%Y %H:%M:%S")

mycomments=f'# ifiles={ifile},ofile={ofile}, generations={generations}, config_dict={config_dict}, EARLY_STOP={EARLY_STOP}\n'
mycomments+=f' # model={model}: {models[model]}, Tpot_file={Tpot_file}, Tpot_score={Tpot_score}, sdata={sdata}, edate={edate}\n'

mmutils.prependComments(Tpot_file,mycomments)
mmutils.prependComments(train_file,mycomments)
mmutils.prependComments(test_file,mycomments)
-- VISUAL LINE --

```

(a) Tpot script on data to find optimal Configuration:100 generations used

```

File Edit View Terminal Tabs Help
Terminal - admin@omen875: ~/MSC/statkep
Optimization Progress: 76% | 7700/10100 [14:10:17<1:46:30, 2.66s/pipeline]
Optimization Progress: 77% | 7800/10100 [14:20:10<1:58:52, 3.10s/pipeline]
Optimization Progress: 78% | 7900/10100 [14:30:19<1:50:19, 3.01s/pipeline]
Optimization Progress: 79% | 8000/10100 [14:43:12<1:59:32, 3.42s/pipeline]
Optimization Progress: 80% | 8100/10100 [14:53:10<1:15:24, 2.26s/pipeline]
Optimization Progress: 81% | 8200/10100 [15:11:36<7:54:59, 15.00s/pipeline]
Optimization Progress: 82% | 8300/10100 [15:29:56<1:43:59, 3.47s/pipeline]
Optimization Progress: 83% | 8400/10100 [15:41:09<1:26:01, 3.04s/pipeline]
Optimization Progress: 84% | 8500/10100 [15:47:02<1:25:48, 3.22s/pipeline]
Optimization Progress: 85% | 8600/10100 [15:59:22<3:49:22, 9.18s/pipeline]
Optimization Progress: 86% | 8700/10100 [16:15:46<2:45:44, 7.10s/pipeline]
Optimization Progress: 87% | 8800/10100 [16:26:35<1:27:01, 4.02s/pipeline]
Optimization Progress: 88% | 8900/10100 [16:39:32<4:25:41, 13.28s/pipeline]
Optimization Progress: 89% | 9000/10100 [16:49:26<1:29:36, 4.89s/pipeline]
Optimization Progress: 90% | 9100/10100 [16:58:20<1:42:51, 6.17s/pipeline]
Optimization Progress: 91% | 9200/10100 [17:13:37<5:46:02, 23.07s/pipeline]
Optimization Progress: 92% | 9300/10100 [17:26:07<5:04:03, 22.80s/pipeline]
Optimization Progress: 93% | 9400/10100 [17:31:38<33:43, 2.89s/pipeline]
Optimization Progress: 94% | 9500/10100 [17:49:39<19:50, 1.98s/pipeline]
Optimization Progress: 95% | 9600/10100 [18:06:34<54:44, 6.57s/pipeline]
Optimization Progress: 96% | 9700/10100 [18:21:50<26:11, 3.93s/pipeline]
Optimization Progress: 97% | 9800/10100 [18:31:36<09:58, 1.99s/pipeline]
Optimization Progress: 98% | 9900/10100 [18:50:31<08:54, 2.67s/pipeline]
Optimization Progress: 99% | 10000/10100 [19:06:54<13:17, 7.97s/pipeline]
Optimization Progress: 100% | 10100/10100 [19:26:30<00:00, 5.44s/pipeline]

Best pipeline: LogisticRegression(Normalizer(GaussianNB(KNeighborsClassifier(KNeighborsClassifier(VarianceThreshold(RobustScaler(BernoulliNB(DecisionTreeClassifier(MinMaxScaler(input_matrix), criterion=gini, max_depth=6, min_samples_leaf=1, min_samples_split=19), alpha=1.0, fit_prior=True)), threshold=0.2), n_neighbors=9, p=1, weights=uniform), n_neighbors=9, p=1, weights=uniform)), norm=l2), C=15.0, dual=False, penalty=l2)
Score see data/Tkb_100g_light_Tpot.py
(base) admin@omen875: ~/MSC/statkep$ 

```

(b) Optimization script duration 19 hours and 26 Minutes

```

File Edit View Terminal Tabs Help
Terminal - admin@omen875: ~/MSC/statkep/data
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import make_pipeline, make_union
from sklearn.preprocessing import MinMaxScaler, Normalizer, RobustScaler
from sklearn.tree import DecisionTreeClassifier
from tpot.builtins import StackingEstimator
from tpot.export_utils import set_param_recursive

# NOTE: Make sure that the outcome column is labeled 'target' in the data file
tpot_data = pd.read_csv('PATH/TO/DATA/FILE', sep='COLUMN_SEPARATOR', dtype=np.float64)
features = tpot_data.drop('target', axis=1)
training_features, testing_features, training_target, testing_target = \
    train_test_split(features, tpot_data['target'], random_state=42)

# Average CV score on the training set was: 0.9317001744979585
exported_pipeline = make_pipeline(
    MinMaxScaler(),
    StackingEstimator(estimator=DecisionTreeClassifier(criterion="gini", max_depth=6, min_samples_leaf=1, min_samples_split=19)),
    StackingEstimator(estimator=BernoulliNB(alpha=1.0, fit_prior=True)),
    RobustScaler(),
    VarianceThreshold(threshold=0.2),
    StackingEstimator(estimator=KNeighborsClassifier(n_neighbors=8, p=1, weights="uniform")),
    StackingEstimator(estimator=KNeighborsClassifier(n_neighbors=9, p=1, weights="uniform")),
    StackingEstimator(estimator=GaussianNB()),
    Normalizer(norm="l2"),
    LogisticRegression(C=15.0, dual=False, penalty="l2"))

# Fix random state for all the steps in exported pipeline
set_param_recursive(exported_pipeline.steps, 'random_state', 42)

exported_pipeline.fit(training_features, training_target)
results = exported_pipeline.predict(testing_features)
-- VISUAL LINE --

```

(c) Optimal Model script automatically generated by tpot

Figure 16: TPOT generates the optimal script for modelling. In this case Logistic Regression see highlighted

9 Assess

Only LR and DT had results with high precision which were not overfitted by TPOT. Due to lack of space in the original report; additional results for performance are shown in section 9.1 and recovery in section 9.2.

Models generated by TPOT are shown in Figure 16 and described in the previous section. These scripts were copied into a class `mmodels.py` for use by the programs `Treat4.py` and `Treat5.py` see Figure 17. Performance results not included in the Technical Report are in sections 9.1.1, 9.1.2, 9.1.3 and 9.1.4.

File Edit View Terminal Tabs Help

```
def XGBdummy(self):
    print("NOT TO be done")

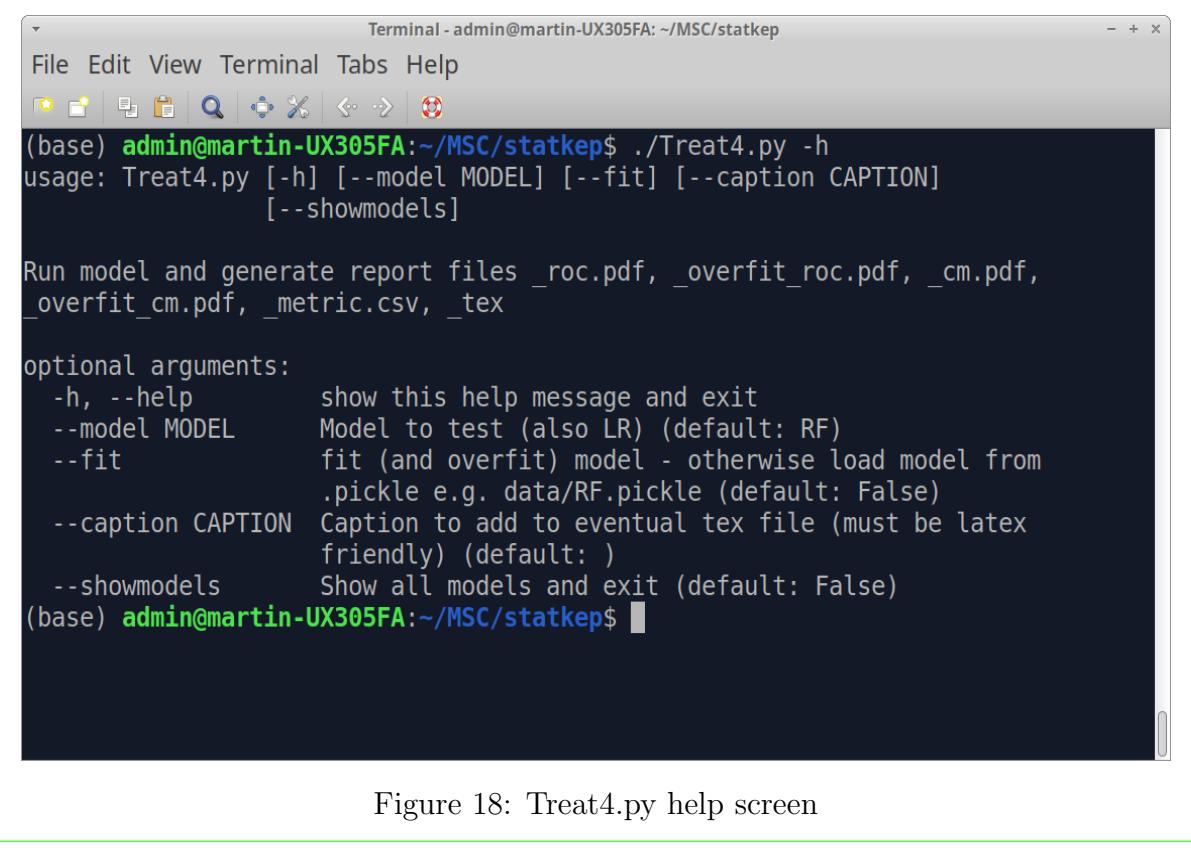
def RF_vif_cap2(self):
    """
    # Average CV score on the training set was: 0.8498227185207794
    # TPOTClassifier(verbosity=2,generations=100,config_dict=None,max_time_mins=1,random_state=42,early_stop=3)# ifile data/TK_vif_cap2.csv model=None Tpot file=data/TK_vif_cap2_100g None Tpot.py, Tpot score=0.3204488778054863, starttime2020-07-20 02:34:14.453438, endtime=2020-07-20 02:35:54.489856 duration=0:1:40
    """
    exported_pipeline = make_pipeline(
        MinMaxScaler(),
        RandomForestClassifier(bootstrap=True, criterion="gini", max_features=0.2, min_samples_leaf=1, min_samples_split=4, n_estimators=100)
    )
    clf=OneVsRestClassifier(exported_pipeline)
    return clf

def DT_vif_cap2(self):
    """
    # Average CV score on the training set was: 0.8042392061782643
    # TPOTClassifier(verbosity=2,generations=100,config_dict=TPOT_light,max_time_mins=1,random_state=42,early_stop=3)# ifile data/TK_vif_cap2.csv model=Tpot_light Tpot_file=data/TK_vif_cap2_100g_Tpot_light_Tpot.py, Tpot score=0.31546134663341646, starttime2020-07-20 02:29:16.873318, endtime=2020-07-20 02:30:18.711808 duration=0:1:1
    """
    exported_pipeline = DecisionTreeClassifier(criterion="entropy", max_depth=8, min_samples_leaf=11, min_samples_split=9)
    # clf = OneVsRestClassifier(exported_pipeline)
    # return clf
    clf=OneVsRestClassifier(exported_pipeline)
    return clf
```

Figure 17: Each TPOT model copied from 16 , assigned unique name and saved as a method in the class mmodels.py

9.1 Performance Evaluation: Treat4.py

Performance evaluation was performed by the program Treat4.py. Help is shown in Figure 18 and the options described in Table 10. The program generated files containing graphs and metrics and a latex report see Figure 19.



The image shows a terminal window titled "Terminal - admin@martin-UX305FA:~/MSC/statkep". The window contains the help output for the "Treat4.py" program. The help text includes usage instructions, optional arguments, and descriptions of each argument. The terminal window has a standard Linux-style interface with a menu bar, icons, and a scroll bar.

```
Terminal - admin@martin-UX305FA:~/MSC/statkep$ ./Treat4.py -h
usage: Treat4.py [-h] [--model MODEL] [--fit] [--caption CAPTION]
                  [--showmodels]

Run model and generate report files _roc.pdf, _overfit_roc.pdf, _cm.pdf,
_overfit_cm.pdf, _metric.csv, _tex

optional arguments:
  -h, --help            show this help message and exit
  --model MODEL         Model to test (also LR) (default: RF)
  --fit                 fit (and overfit) model - otherwise load model from
                        .pickle e.g. data/RF.pickle (default: False)
  --caption CAPTION    Caption to add to eventual tex file (must be latex
                        friendly) (default: )
  --showmodels          Show all models and exit (default: False)
(base) admin@martin-UX305FA:~/MSC/statkep$
```

Figure 18: Treat4.py help screen

Table 10: Summary of program Treat4.py

Modification	Reason
model	The tpot model to be run
fit	By default the model will be loaded from a pickle file but if fit option is called the model will be called and create the pickle file.
caption	The model created will generate ROC curves and metrics. This option will add a caption
showmodels	Show models available

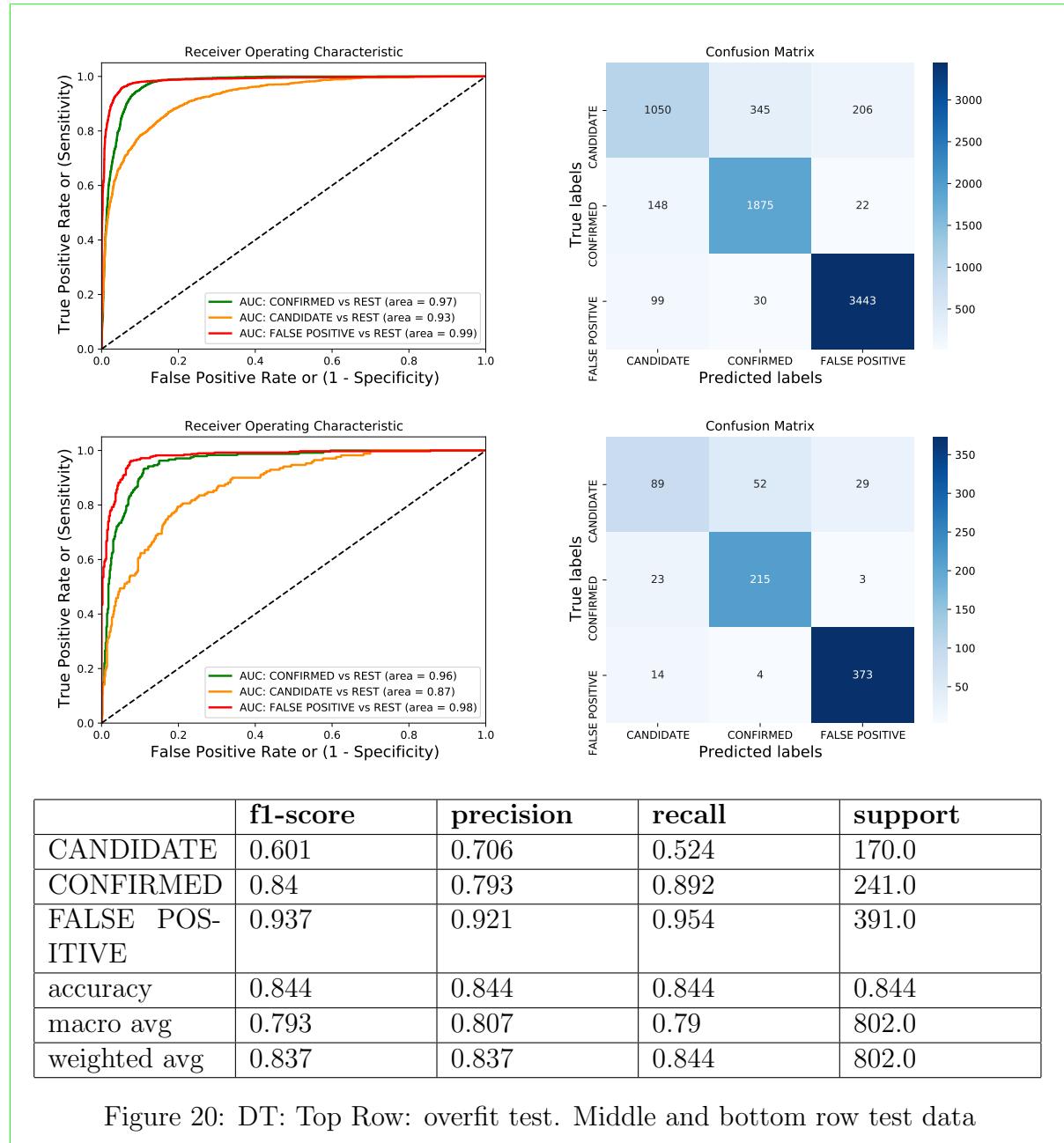
```
File Edit View Terminal Tabs Help
File Edit View Terminal Tabs Help
\expandafter \newcommand\c{dataRFmetricab}\endc
\begin{table}[H]
\begin{tabular}
{|
p{\dimexpr0.2\textwidth-2\tabcolsep-\arrayrulewidth\relax}|
p{\dimexpr0.2\textwidth-2\tabcolsep-\arrayrulewidth\relax}|
p{\dimexpr0.2\textwidth-2\tabcolsep-\arrayrulewidth\relax}|
p{\dimexpr0.2\textwidth-2\tabcolsep-\arrayrulewidth\relax}|
p{\dimexpr0.2\textwidth-2\tabcolsep-\arrayrulewidth\relax}|
}
\hline
\textbf{f1-score} & \textbf{precision} & \textbf{recall} & \textbf{support} \\ \hline
CANDIDATE & 0.635 & 0.756 & 0.547 & 170.0 \\ \hline
CONFIRMED & 0.849 & 0.806 & 0.896 & 241.0 \\ \hline
FALSE POSITIVE & 0.94 & 0.917 & 0.964 & 391.0 \\ \hline
accuracy & 0.855 & 0.855 & 0.855 & 0.855 \\ \hline
macro avg & 0.808 & 0.826 & 0.803 & 0.802.0 \\ \hline
weighted avg & 0.848 & 0.85 & 0.855 & 0.802.0 \\ \hline
\end{tabular}
\end{table}
}
\begin{figure}[H]
\centering
\begin{subfigure}{.49\textwidth}
\includegraphics[width = 1\textwidth]{data/RF_overfit_roc.pdf}
\end{subfigure}
\begin{subfigure}{.49\textwidth}
\includegraphics[width = 1\textwidth]{data/RF_overfit_cm.pdf}
\end{subfigure}
\begin{subfigure}{.49\textwidth}
\includegraphics[width = 1\textwidth]{data/RF_roc.pdf}
\end{subfigure}
\end{figure}

```

Figure 19: Treat4.py generates ROC pdf files using matplotlib and these are put into a file in latex format see Figure 30

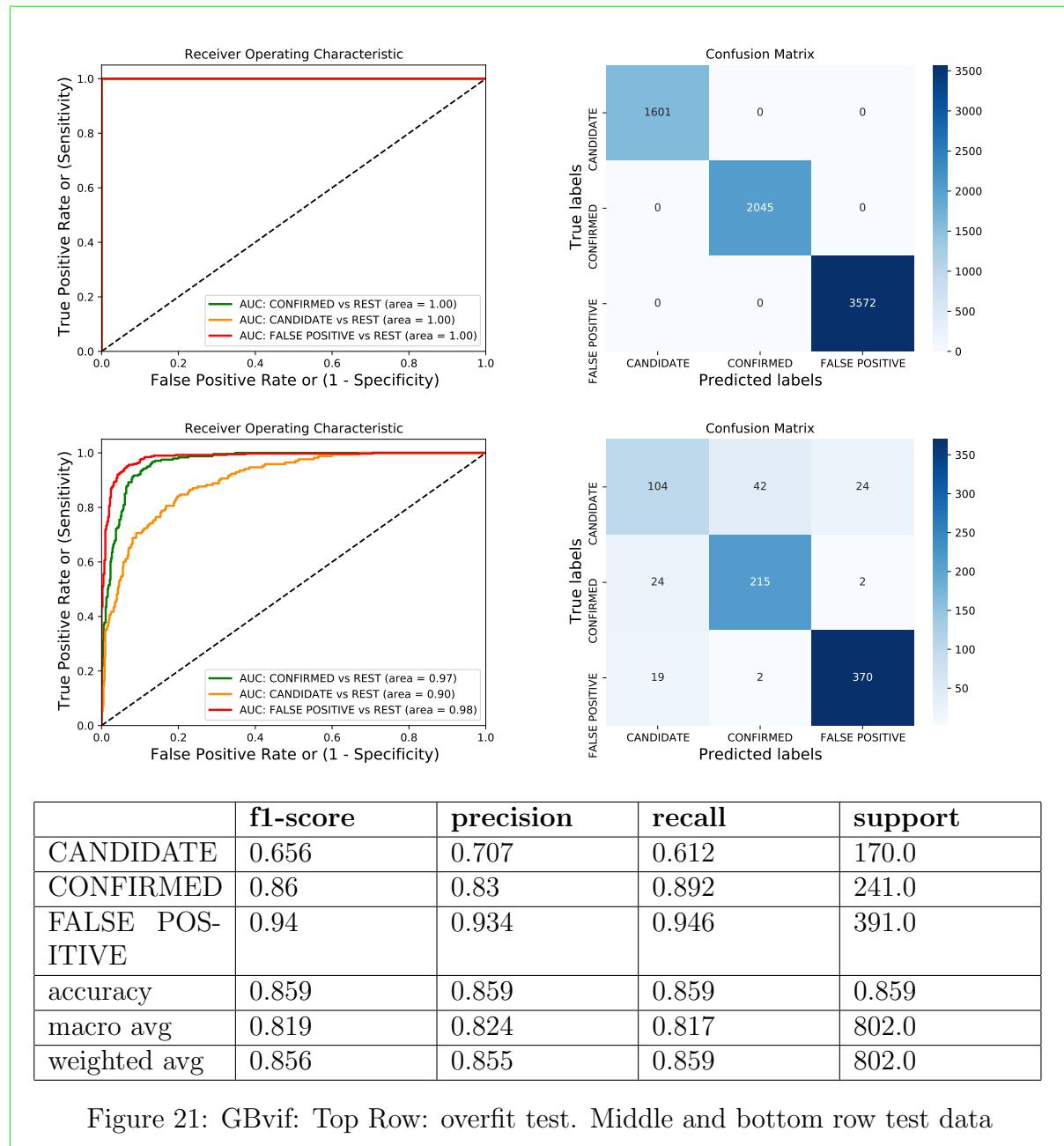
9.1.1 Decision Tree

The Decision Tree performance is summarized in Figure 20.



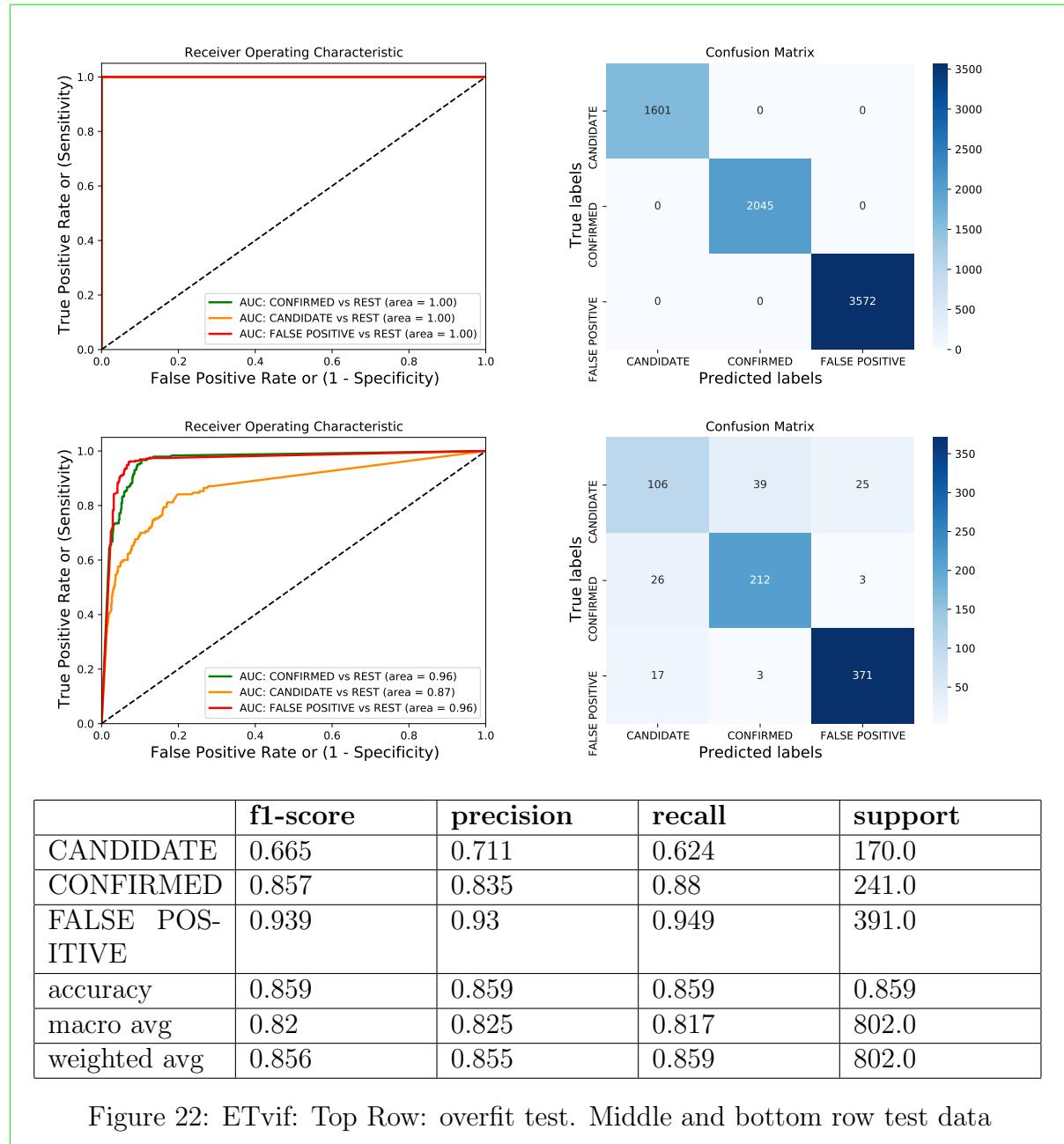
9.1.2 Gradient Boost

The Gradient Boost performance is summarized in Figure 21. 100% recovery is obtained with training data indicating overfitting.



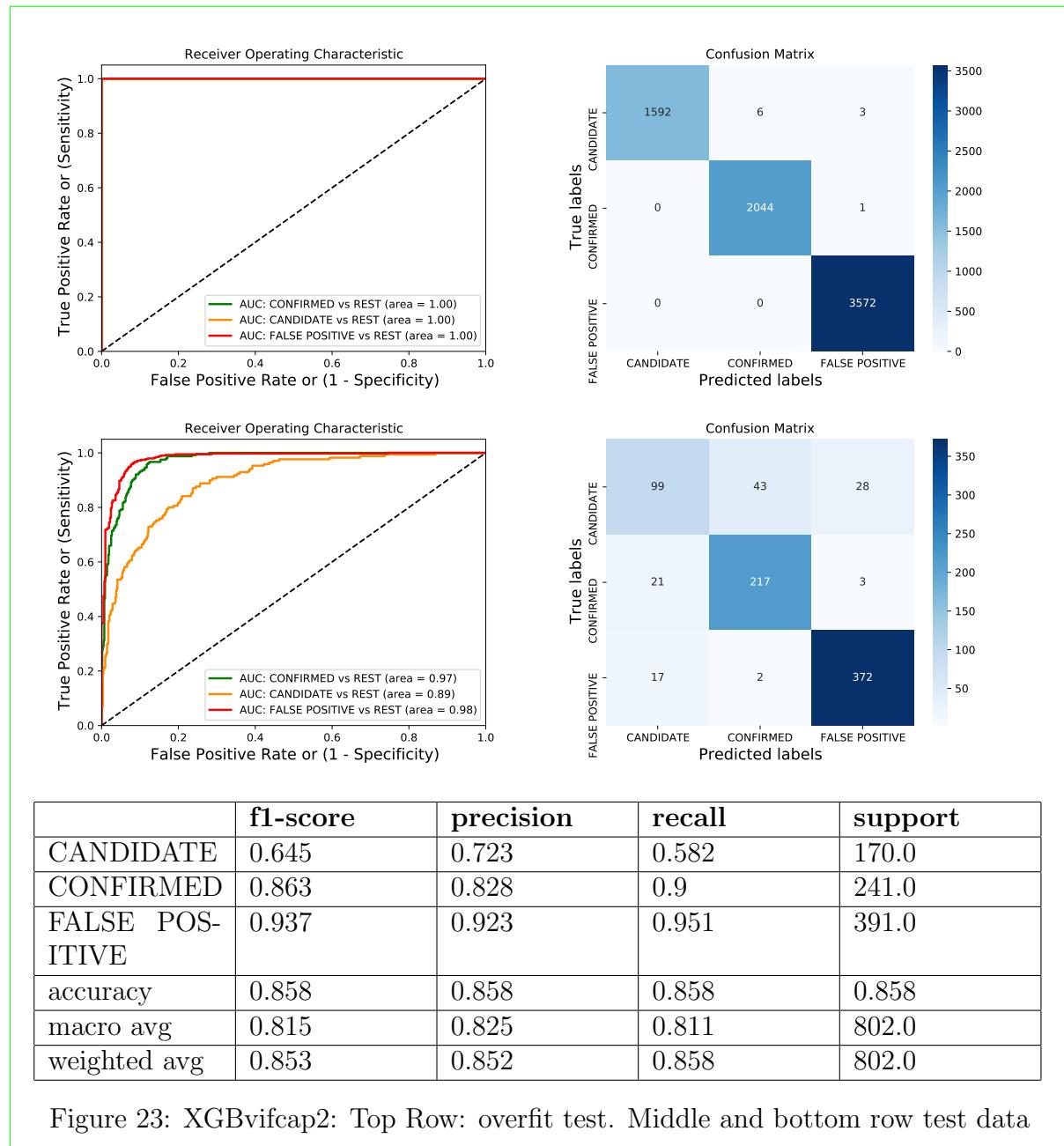
9.1.3 Extra Tree Classifier

The Extra Tree Classifier performance is summarized in Figure 22. 100% recovery is obtained with training data indicating overfitting.



9.1.4 Xtreme Gradient Boost

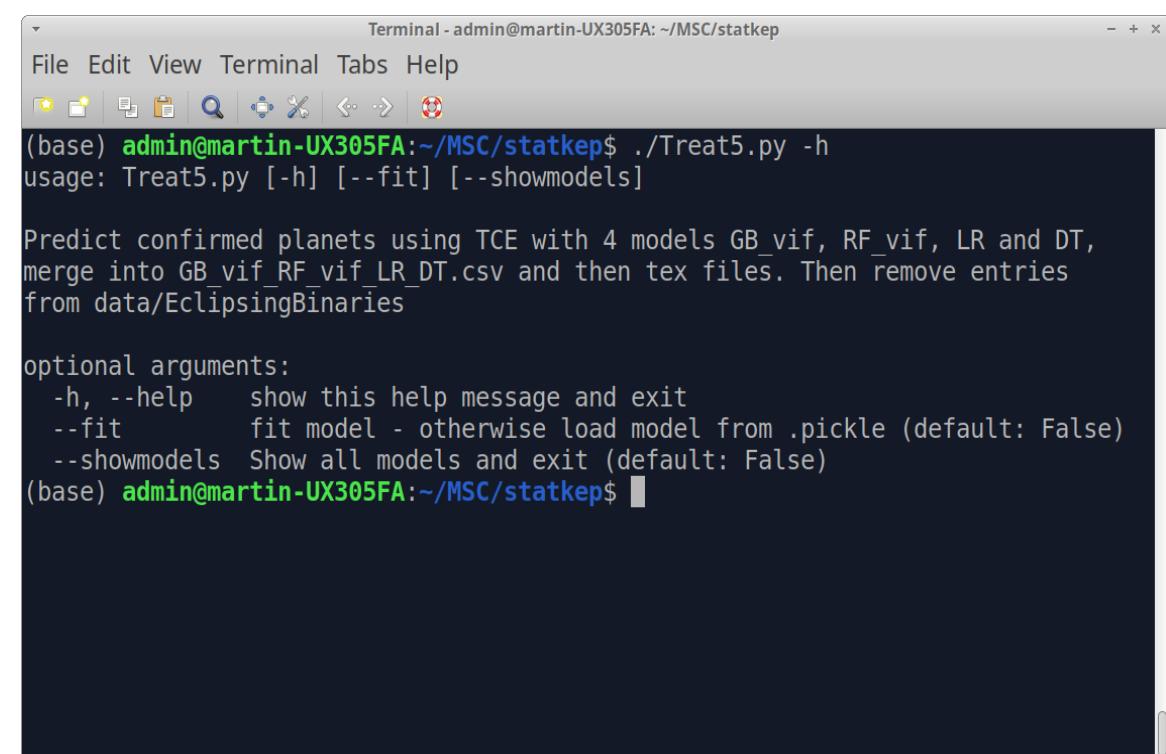
The Xtreme Gradient Boost performance is summarized in 100% recovery is obtained with training data indicating overfitting.



9.2 Recovery: Treat5.py

Help for Treat5.py is shown in Figure 24 and the options are summarized in Table 11. After recovery eclipsing binaries were removed.⁸. Graphs were generated using matplotlib and a latex report was generated using these graphs see 9.2.1 and 9.2.2. Recovery results not included in the Technical Report are in sections 9.2.1, 9.2.2, 9.2.3 and 9.2.4.

A full list of results comparing four different models was then generated see Table 12 and Table 13



The image shows a terminal window titled "Terminal - admin@martin-UX305FA: ~/MSC/statkep". The window contains the help output for the "Treat5.py" program. The output includes the usage command, a detailed description of the program's purpose, and a list of optional arguments with their descriptions. The terminal window has a standard OS X-style interface with a menu bar and a toolbar.

```
File Edit View Terminal Tabs Help
File Finder Applications Spotlight System Preferences
(base) admin@martin-UX305FA:~/MSC/statkep$ ./Treat5.py -h
usage: Treat5.py [-h] [--fit] [--showmodels]

Predict confirmed planets using TCE with 4 models GB_vif, RF_vif, LR and DT,
merge into GB_vif_RF_vif_LR_DT.csv and then tex files. Then remove entries
from data/EclipsingBinaries

optional arguments:
-h, --help    show this help message and exit
--fit        fit model - otherwise load model from .pickle (default: False)
--showmodels Show all models and exit (default: False)
(base) admin@martin-UX305FA:~/MSC/statkep$
```

Figure 24: Treat5.py help screen

Table 11: Summary of program Treat5.py

Modification	Reason
fit	By default the model will be loaded from a pickle file but if fit option is called the model will be called and create the pickle file.
showmodels	Show models available
model	Confirmed planets for individual models can be recovered

⁸http://archive.stsci.edu/kepler/eclipsing_binaries.html

9.2.1 Decision Tree

The confirmed planets recovered by Decision Tree are shown in Figure 25.

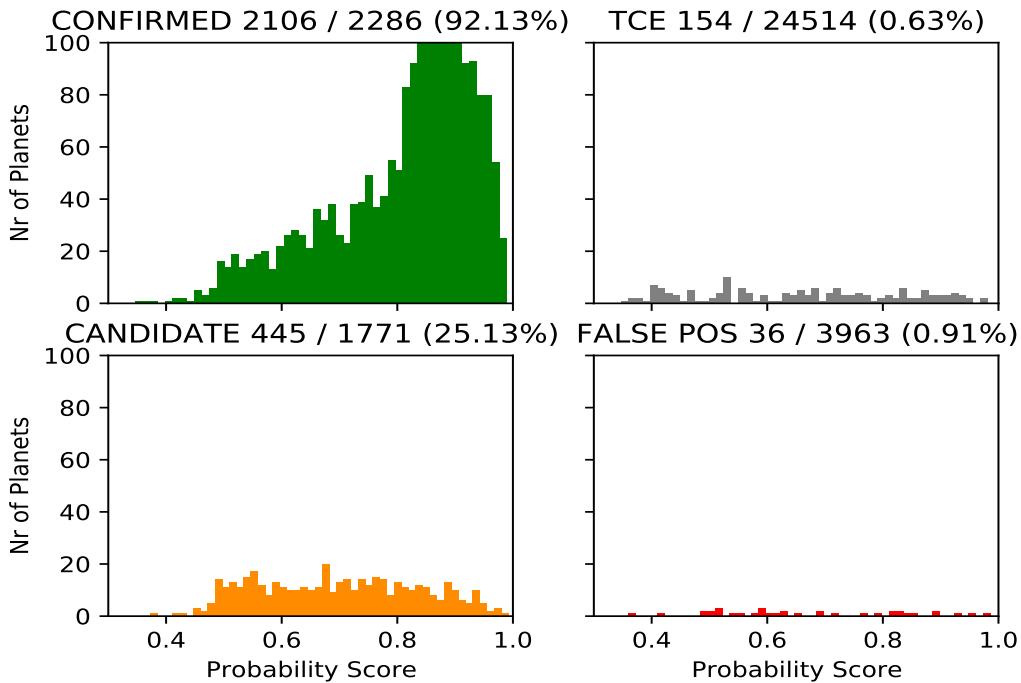
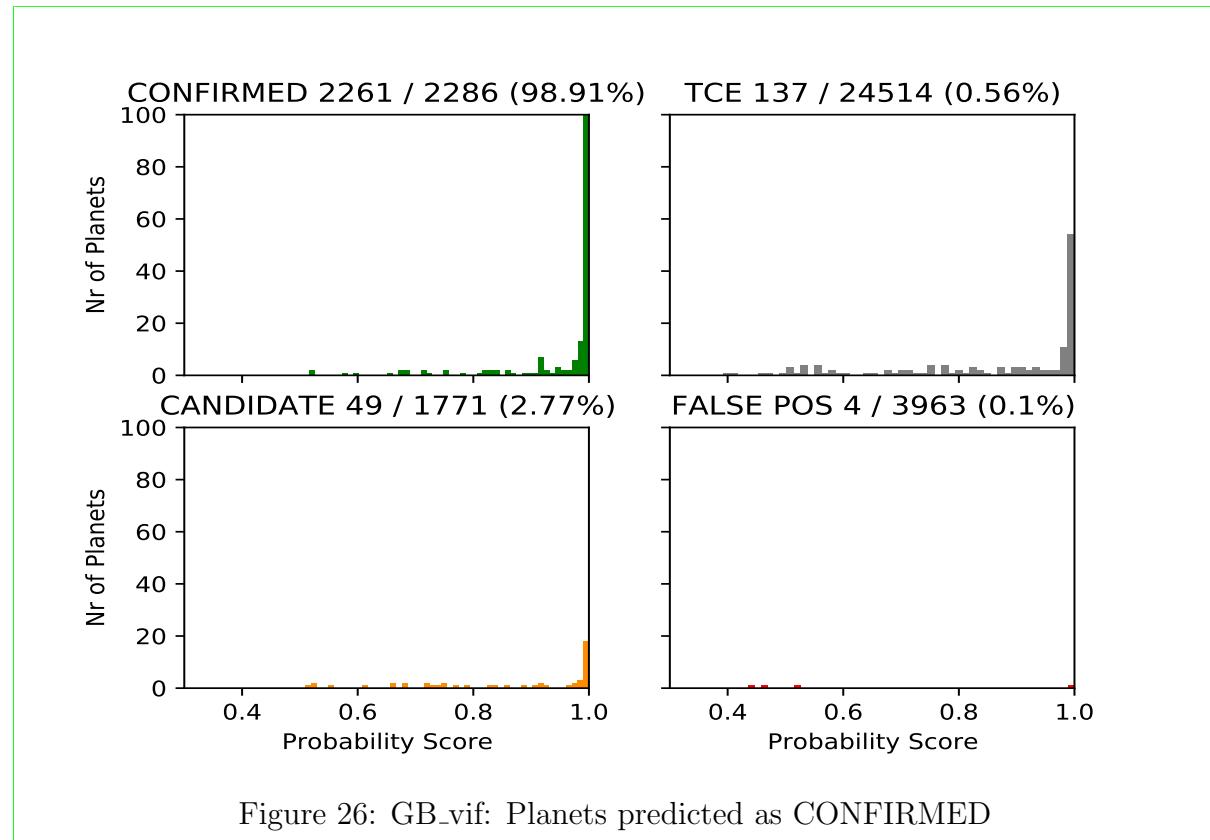


Figure 25: DT: Planets predicted as CONFIRMED

9.2.2 Gradient Boost

The confirmed planets recovered by Gradient Boost are shown in Figure 26.



9.2.3 Extra Tree Classifier

The confirmed planets recovered by Extra Tree Classifier are shown in Figure 27.

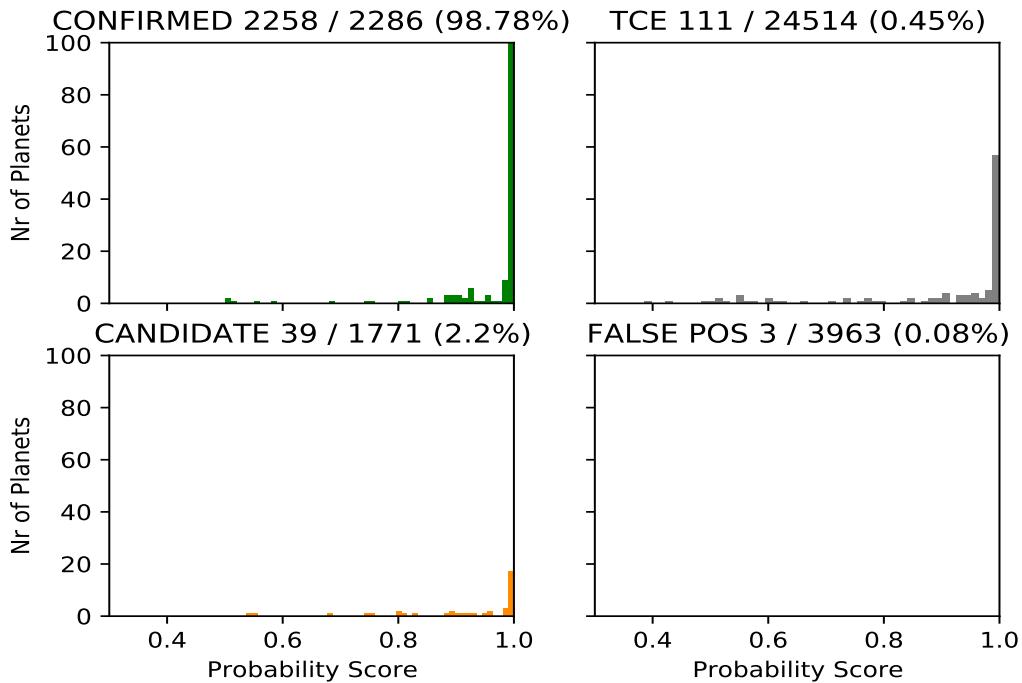
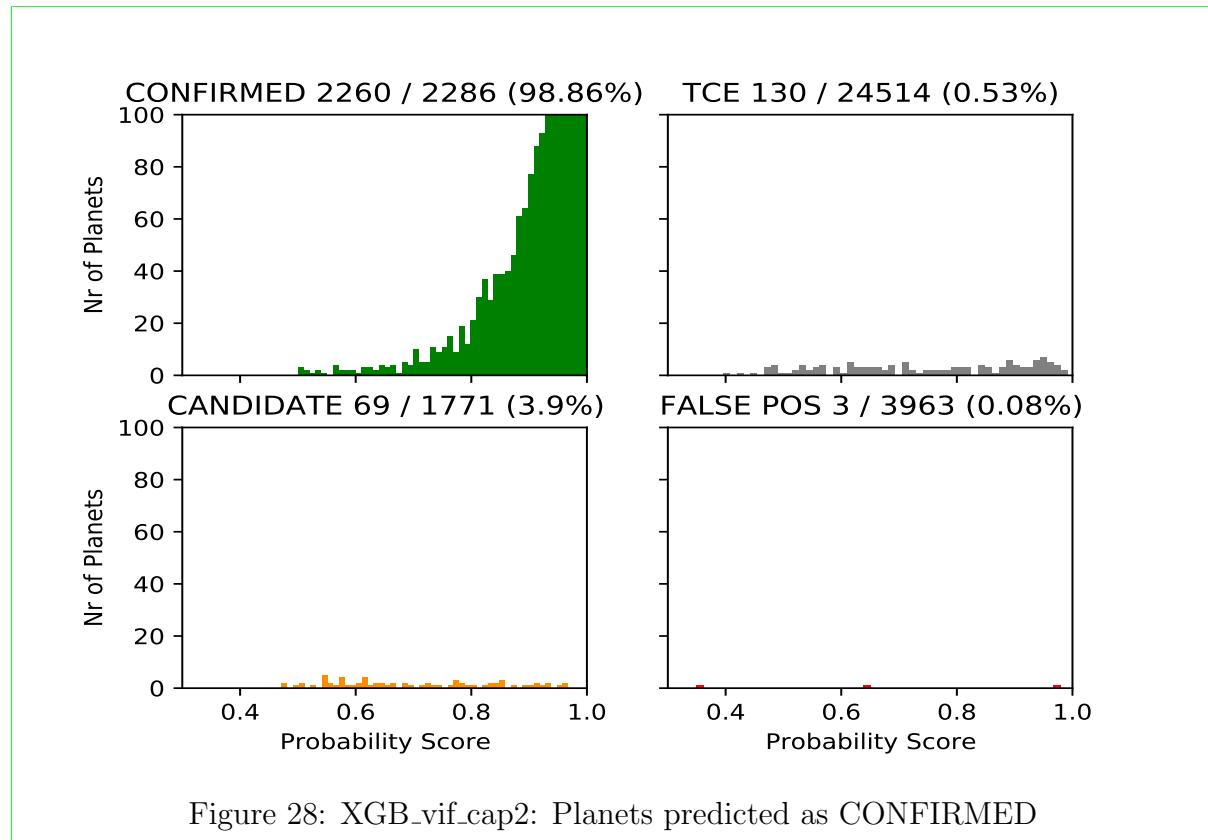


Figure 27: ET_vif: Planets predicted as CONFIRMED

9.2.4 Xtreme Gradient Boost

The confirmed planets recovered by Xtreme Gradient Boost are shown in Figure 28.



9.3 List of recovered planets ordered by Logistic Regression (LR) with eclipsing binaries

Table 12 shows the full list of planets. This table is ordered by Logistic Regression and Eclipsing binaries are marked with 1 under eb . The table illustrates that LR and DT mainly return planets from the KOI table with high probability.

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

kepid	plnt num	kname	dispos	LR	DT	GB_vif	RF_vif	eb
8456679	2	K00102.02	FP	0.972	0.951	-	-	0
8480285	1	K00691.01	CAND	0.971	0.948	-	0.503	0
8804455	2	K02159.02	FP	0.963	0.986	-	-	0
10788461	1	K03925.01	CAND	0.959	0.571	-	-	0
9071386	2	-	-	0.956	0.942	1.0	0.843	1
7037540	2	-	-	0.956	0.889	0.999	0.98	1
8552719	2	K01792.02	FP	0.949	0.888	1.0	0.98	1
8644365	1	K03384.01	CAND	0.948	0.943	-	-	0
8804845	1	K02039.01	CAND	0.945	0.919	-	-	0
3831053	1	K00388.01	CAND	0.944	0.966	-	0.642	0
12505076	1	K02154.01	CAND	0.944	0.971	-	-	0
4149450	1	K01864.01	CAND	0.943	0.992	-	-	0
2581316	2	K03681.02	CAND	0.942	0.932	1.0	0.952	0
5709725	2	K00555.02	CAND	0.937	0.919	-	0.516	0
3970233	2	-	-	0.935	0.972	0.999	0.97	1
5374854	2	K00645.02	CAND	0.93	0.94	-	-	0
9838975	2	-	-	0.927	0.879	1.0	0.99	1
8180020	2	-	-	0.927	0.925	0.999	0.979	1
3247268	2	K01089.02	CAND	0.927	0.886	-	-	0
11098013	1	K02712.01	CAND	0.924	0.939	-	-	0
9411166	2	-	-	0.921	0.684	0.974	0.979	0
3641726	1	K00804.01	CAND	0.917	0.847	-	-	0
7938496	1	K00900.01	CAND	0.916	0.853	-	-	0
6364215	2	K02404.02	FP	0.915	0.82	-	-	0
5972334	2	K00191.02	CAND	0.915	0.933	-	-	0
10794087	1	K01316.01	CAND	0.914	0.899	1.0	0.959	0
8934495	1	K00524.01	CAND	0.91	0.689	-	-	0
11566064	2	K00353.02	CAND	0.909	0.809	-	-	0
9446824	2	-	-	0.908	0.913	0.999	0.979	1
11030475	1	K02248.01	CAND	0.907	0.875	-	-	0
8176650	2	-	-	0.905	0.874	1.0	0.971	0
9605514	1	K00945.01	CAND	0.905	0.893	-	-	0
12504988	2	-	-	0.905	0.921	1.0	0.969	1
3342592	2	-	-	0.904	0.935	0.998	0.949	1
9605514	2	K00945.02	CAND	0.903	0.85	-	-	0
5965819	1	K03319.01	CAND	0.903	0.89	-	0.508	1

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

5531953	1	K01681.01	CAND	0.903	0.844	-	-	0
5972334	3	K00191.03	CAND	0.9	0.935	-	-	0
3642741	1	K00242.01	FP	0.899	0.852	-	-	0
10867062	1	K01303.01	CAND	0.896	0.901	-	-	0
5794570	2	K02675.02	CAND	0.896	0.764	-	-	0
9823487	1	K01489.01	CAND	0.894	0.882	-	-	0
11187837	1	K00252.01	CAND	0.893	0.862	-	-	1
8495415	1	K02362.01	FP	0.892	-	-	-	1
6762829	2	-	-	0.892	0.839	0.999	0.96	1
9266431	1	K00704.01	CAND	0.89	0.9	-	-	0
11071200	2	K02696.02	CAND	0.887	0.858	-	-	0
5770074	1	K01928.01	CAND	0.887	0.719	-	-	0
2853780	1	K02081.01	CAND	0.885	0.943	-	-	0
5792202	3	K00841.03	CAND	0.884	0.832	-	-	0
9150827	1	K01408.01	CAND	0.883	0.865	-	0.505	0
4851464	2	-	-	0.883	0.923	0.999	0.929	1
8613535	1	K02263.01	CAND	0.88	0.907	-	-	0
5031857	2	K01573.02	CAND	0.879	0.584	-	0.967	0
5374838	1	K05155.01	CAND	0.877	0.66	-	-	0
5342473	1	K02297.01	CAND	0.875	0.818	-	-	0
8073806	1	K02411.01	CAND	0.873	0.899	-	-	0
5211199	1	K02158.01	CAND	0.873	0.92	-	-	0
11297236	1	K01857.01	CAND	0.872	0.686	-	-	0
5031857	1	K01573.01	CAND	0.871	0.676	-	-	0
7697568	1	K01829.01	FP	0.869	-	-	-	0
9006186	2	K02169.02	CAND	0.868	0.886	-	-	0
11874577	1	K02779.01	CAND	0.868	0.845	-	-	0
2438513	1	K01944.01	CAND	0.867	0.952	-	-	0
9592705	1	K00288.01	CAND	0.867	0.734	-	-	1
6049190	1	K01685.01	CAND	0.867	0.564	-	-	0
11709124	2	-	-	0.867	0.938	1.0	0.918	0
5035972	2	-	-	0.866	0.561	0.75	0.903	1
5370302	2	-	-	0.866	0.908	1.0	0.971	1
5020319	1	K00635.01	CAND	0.866	0.912	-	0.508	0
3940418	1	K00810.01	CAND	0.866	0.824	-	-	0
11404644	2	-	-	0.865	0.956	0.962	-	1
6946199	2	K01359.02	CAND	0.865	0.759	-	-	0
7670943	1	K00269.01	CAND	0.864	0.904	-	-	0
2449431	1	K02009.01	CAND	0.861	0.863	0.984	0.931	0
10710755	2	-	-	0.86	0.903	0.716	0.389	1
10676824	1	K00599.01	CAND	0.859	0.77	-	-	0
5786676	1	K00650.01	CAND	0.858	0.811	0.999	0.949	0
9632895	2	-	-	0.856	0.742	1.0	0.462	1
11810124	1	K03344.01	CAND	0.854	0.761	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

6383785	2	K00239.02	CAND	0.853	0.853	-	-	0
5195172	1	K01671.01	CAND	0.85	0.978	-	-	0
10134152	1	K02056.01	CAND	0.85	0.906	-	-	0
10016874	1	K00426.01	CAND	0.85	0.809	0.717	0.96	0
11071200	1	K02696.01	CAND	0.848	0.633	-	-	0
10904857	2	-	-	0.845	0.879	0.994	0.96	1
9117416	2	K03425.02	CAND	0.845	0.841	-	-	1
11408935	2	-	-	0.843	0.898	0.994	0.915	1
4770365	2	K01475.02	CAND	0.843	0.785	-	-	0
7887791	1	K01964.01	CAND	0.841	-	-	-	0
7364176	1	K00373.01	CAND	0.84	0.679	-	-	0
9006186	3	K02169.03	CAND	0.838	0.893	-	-	0
9904006	2	K02135.02	CAND	0.837	0.746	-	-	0
3097926	1	K02186.01	CAND	0.837	0.804	0.985	0.959	0
11026304	1	K01420.01	CAND	0.836	-	-	-	0
3128552	3	K02055.03	CAND	0.834	0.77	-	-	0
11337141	1	K01649.01	CAND	0.832	-	-	-	0
7899070	1	K02683.01	CAND	0.832	0.838	-	-	0
5534941	1	K02332.01	CAND	0.83	0.969	1.0	0.929	0
4365645	1	K01738.01	CAND	0.83	0.884	-	-	0
9244756	1	K01969.01	CAND	0.83	0.848	-	-	0
10395543	1	K00531.01	CAND	0.829	0.907	-	-	1
9636135	2	K01498.02	CAND	0.828	-	-	-	0
6587002	3	K00612.03	CAND	0.826	0.855	-	-	0
6698670	2	-	-	0.826	0.84	0.997	0.849	1
7948784	1	K01968.01	CAND	0.823	0.708	-	-	0
3102000	2	-	-	0.822	0.913	0.999	0.898	1
11705004	1	K02199.01	CAND	0.822	0.544	-	-	0
9837661	3	K02715.03	CAND	0.82	0.75	-	-	0
8460600	2	-	-	0.82	0.817	0.992	0.565	1
12017109	1	K02106.01	CAND	0.819	0.848	-	-	0
11599038	1	K01437.01	CAND	0.819	0.896	-	-	0
8263545	1	K02822.01	CAND	0.819	0.888	0.991	0.939	0
5376836	2	-	-	0.819	0.869	0.999	0.98	1
9288786	2	-	-	0.819	0.896	1.0	0.909	1
4466677	2	K01338.02	CAND	0.817	0.526	-	-	0
11030475	2	-	-	0.816	0.736	0.999	0.907	0
10485250	2	-	-	0.814	0.761	-	0.538	1
3109930	1	K01112.01	CAND	0.814	0.795	-	-	0
5219234	3	K01563.03	CAND	0.813	0.644	-	-	0
11852982	1	K00247.01	CAND	0.812	0.838	-	-	0
7256914	1	K04136.01	CAND	0.811	0.949	-	-	0
9649222	2	-	-	0.81	0.862	0.988	0.915	1
8892910	2	-	-	0.809	0.591	0.998	0.978	1

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

5384713	1	K03444.01	CAND	0.809	-	-	-	1
11568987	2	K00354.02	CAND	0.805	0.741	-	-	1
7584650	1	K02631.01	CAND	0.805	0.885	1.0	0.96	0
8686150	2	-	-	0.805	0.89	0.935	0.643	1
7679812	1	K03943.01	CAND	0.804	0.864	-	-	0
9896018	2	K02579.02	CAND	0.804	0.726	-	-	0
7975727	2	-	-	0.804	0.852	0.999	0.96	1
9697131	1	K02706.01	CAND	0.803	0.682	0.679	0.736	0
5042210	1	K02462.01	CAND	0.803	0.967	0.92	0.918	0
8043714	2	-	-	0.803	0.838	-	0.929	1
11752906	1	K00253.01	CAND	0.803	0.795	-	-	0
9532710	1	K02364.01	FP	0.802	0.716	-	-	0
11853878	2	K01833.02	CAND	0.802	-	-	-	0
9896018	1	K02579.01	CAND	0.8	0.887	-	-	0
7386827	1	K01704.01	CAND	0.799	0.81	-	-	0
8078502	1	K03383.01	CAND	0.798	0.923	-	-	0
7023960	2	-	-	0.798	0.859	0.996	0.957	1
4164922	1	K03864.01	CAND	0.797	0.668	-	-	0
8552202	1	K00914.01	CAND	0.796	-	-	-	0
10287248	2	-	-	0.795	0.892	0.995	0.84	1
12120943	1	K01458.01	CAND	0.793	0.748	-	-	0
11821363	1	K01494.01	CAND	0.792	0.8	-	-	0
10587105	3	K00339.03	CAND	0.791	0.55	-	-	0
7983117	1	K03214.01	CAND	0.791	-	-	-	0
5780885	2	-	-	0.787	0.607	0.912	0.4	0
9405595	1	K02125.01	CAND	0.787	0.822	-	-	0
10713616	1	K01311.01	CAND	0.784	0.648	-	-	0
5601258	1	K02191.01	CAND	0.784	0.8	-	-	0
8558011	1	K00577.01	CAND	0.784	0.777	-	-	0
9595686	1	K00944.01	CAND	0.783	0.767	-	-	0
3663173	1	K02750.01	CAND	0.783	0.616	-	-	0
10875007	1	K04149.01	CAND	0.781	0.76	-	-	0
9269281	1	K01398.01	CAND	0.781	-	-	-	0
3734868	2	-	-	0.778	0.643	1.0	0.99	1
8578780	1	K02023.01	CAND	0.778	0.888	-	-	0
10091110	3	-	-	0.778	0.742	0.999	0.947	1
8505920	1	K02094.01	CAND	0.777	0.835	-	-	0
11126381	1	K01863.01	CAND	0.776	0.889	0.999	0.933	0
11030711	1	K01429.01	CAND	0.776	0.585	-	-	0
11074178	2	K01889.02	CAND	0.775	0.896	-	-	0
5356593	2	-	-	0.775	0.839	0.77	0.948	1
9579641	3	K00115.03	CAND	0.774	-	-	-	0
4770365	1	K01475.01	CAND	0.773	0.832	-	-	0
7733731	1	K04022.01	CAND	0.771	0.944	-	-	1

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

9267996	1	K03431.01	FP	0.771	-	-	-	0
8458207	2	-	-	0.77	0.758	0.979	-	1
5953297	1	K02733.01	CAND	0.767	0.865	-	-	0
12469800	1	K02543.01	CAND	0.764	0.679	-	-	0
5872150	3	-	-	0.764	0.777	0.99	0.922	1
6198182	1	K02636.01	CAND	0.764	0.782	-	-	0
4242147	1	K01934.01	CAND	0.761	0.803	0.909	0.929	0
4951877	1	K00501.01	CAND	0.761	0.841	-	-	0
10877367	2	-	-	0.758	-	-	-	0
9517393	1	K02076.01	CAND	0.758	0.491	-	-	0
9772531	1	K00950.01	CAND	0.757	0.77	-	-	1
10350571	2	K01175.02	CAND	0.757	0.785	-	-	0
5629353	1	K06132.01	CAND	0.755	-	-	-	1
8261920	3	K02174.03	CAND	0.755	0.549	-	-	0
11253711	1	K01972.01	CAND	0.754	0.785	-	-	0
7202957	2	K02687.02	CAND	0.754	0.952	-	-	0
6467363	1	K02840.01	CAND	0.753	0.682	-	-	0
4756776	2	-	-	0.752	0.441	-	-	0
11967788	1	K04021.01	CAND	0.752	0.75	-	-	0
3425851	1	K00268.01	CAND	0.752	0.851	-	-	0
11656840	2	-	-	0.75	-	-	-	1
8780959	2	K03741.02	FP	0.75	0.831	-	-	0
10294509	1	K04143.01	FP	0.75	-	-	-	0
11657614	1	K03370.01	CAND	0.749	0.758	-	-	0
6058816	1	K03500.01	CAND	0.749	0.836	-	-	0
5978361	1	K00558.01	CAND	0.746	0.738	-	-	0
7877978	1	K02760.01	CAND	0.745	0.809	-	-	0
10793172	2	K02871.02	CAND	0.743	0.723	-	-	0
11026582	1	K05854.01	CAND	0.742	-	-	-	0
2854181	1	K02232.01	CAND	0.742	0.821	-	-	0
5384713	2	K03444.02	FP	0.741	0.69	-	-	1
12120307	3	K02597.03	CAND	0.741	0.618	-	-	0
7135852	1	K00875.01	CAND	0.74	0.767	-	-	0
9230021	1	K03429.01	CAND	0.739	0.735	-	-	0
5384713	3	K03444.03	CAND	0.736	0.661	-	-	1
8678594	2	K01261.02	CAND	0.736	0.863	-	-	0
3098810	1	K01878.01	CAND	0.733	0.584	-	-	0
10616571	3	-	-	0.731	0.724	1.0	0.96	1
6342333	1	K02065.01	CAND	0.728	0.733	-	-	0
6666233	2	-	-	0.727	0.837	1.0	0.846	0
9729691	2	K01751.02	FP	0.726	-	-	-	0
3859628	1	K01191.01	CAND	0.724	0.788	-	-	0
7449554	1	K02357.01	CAND	0.723	0.79	0.92	0.863	0
10793172	1	K02871.01	CAND	0.721	0.758	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

10319385	1	K01169.01	CAND	0.72	0.751	-	-	0
5787131	1	K02927.01	CAND	0.72	0.766	-	-	0
8564976	1	K03890.01	CAND	0.716	-	-	-	0
6803202	1	K00177.01	CAND	0.714	0.91	-	0.503	0
4860678	1	K01602.01	CAND	0.714	0.741	-	-	0
7183745	2	K02521.02	CAND	0.714	0.855	-	-	0
7287415	1	K01369.01	CAND	0.713	0.937	-	-	0
11718144	1	K02310.01	CAND	0.713	0.833	-	-	0
6719086	1	K01357.01	CAND	0.713	0.715	-	-	0
11244682	1	K03933.01	CAND	0.712	0.882	-	-	0
9414417	1	K00974.01	CAND	0.712	-	-	-	0
10513530	2	K00533.02	CAND	0.712	-	-	-	0
10923260	2	-	-	0.711	0.656	0.979	0.556	1
12785320	2	K00298.02	CAND	0.71	0.729	-	-	0
10973664	2	K00601.02	CAND	0.709	-	-	-	0
9030537	1	K01892.01	CAND	0.707	0.545	-	-	0
4857213	1	K02120.01	CAND	0.706	0.74	-	-	0
4548098	1	K04157.01	CAND	0.705	0.831	-	-	0
3761319	1	K06104.01	CAND	0.704	0.689	-	-	0
5716932	1	K02358.01	CAND	0.703	0.778	-	-	0
10252275	2	K03130.02	CAND	0.703	0.646	-	-	0
12105051	1	K00141.01	CAND	0.703	0.762	-	-	0
7941200	1	K00092.01	CAND	0.702	-	-	-	0
11754430	2	K03403.02	CAND	0.7	0.699	-	-	0
10337517	1	K01165.01	CAND	0.697	0.88	-	0.516	0
2446113	1	K00379.01	FP	0.697	0.927	-	-	0
12156347	1	K02588.01	CAND	0.696	0.718	-	-	0
5542466	1	K01590.01	CAND	0.695	-	-	-	0
6614926	2	-	-	0.693	0.718	0.778	-	1
5978559	1	K02321.01	CAND	0.693	0.698	0.992	0.959	0
10118816	1	K01085.01	CAND	0.693	0.826	-	-	0
7287683	1	K01622.01	CAND	0.692	0.715	-	-	0
11752906	2	K00253.02	CAND	0.691	0.712	-	-	0
8257205	2	K01986.02	CAND	0.691	0.727	-	-	0
12121570	1	K02290.01	CAND	0.691	0.685	-	-	0
8655354	2	-	-	0.691	0.726	0.736	0.5	0
9836959	1	K01715.01	CAND	0.69	0.55	0.977	0.938	0
9934208	2	-	-	0.69	0.876	1.0	0.853	1
10055126	3	K01608.03	CAND	0.689	0.534	-	-	0
7294743	1	K02516.01	CAND	0.688	0.874	-	-	0
5531953	2	K01681.02	FP	0.688	0.833	-	-	0
10387742	1	K02583.01	CAND	0.688	0.9	-	-	0
11599264	2	-	-	0.688	-	-	-	1
7522911	1	K01705.01	CAND	0.688	0.707	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

6956216	2	-	-	0.687	-	-	-	1
12061238	1	K01502.01	FP	0.687	-	-	-	0
10514430	3	-	-	0.685	0.726	-	0.385	0
6975129	3	K01628.03	FP	0.685	0.586	-	-	0
4862625	2	-	-	0.685	-	1.0	0.429	1
7362696	2	-	-	0.685	0.766	0.888	0.895	1
9700449	1	K04120.01	CAND	0.684	-	-	-	0
9489524	4	K02029.04	CAND	0.683	0.489	-	-	0
10363115	1	K03908.01	CAND	0.682	0.781	-	-	0
11138155	1	K00760.01	CAND	0.681	0.656	-	-	1
8494542	1	K02204.01	CAND	0.679	0.679	-	-	0
10386922	1	K00289.01	CAND	0.679	0.631	-	-	0
11771430	1	K02582.01	CAND	0.679	0.689	-	-	0
3323887	2	-	-	0.677	0.714	0.805	-	0
9692128	1	K02769.01	FP	0.677	-	-	-	0
10418797	2	-	-	0.677	0.713	1.0	0.922	1
11462341	1	K02124.01	CAND	0.677	0.686	-	-	0
11442465	2	-	-	0.677	-	-	-	1
8018547	1	K00902.01	CAND	0.676	0.742	-	-	0
6543893	2	K01627.02	CAND	0.674	0.754	-	-	0
8074805	2	-	-	0.674	0.633	0.999	0.791	1
6609270	1	K03090.01	CAND	0.674	0.727	-	-	0
11192235	1	K02329.01	CAND	0.673	0.599	-	0.888	0
9489524	3	K02029.03	CAND	0.673	-	-	-	0
5283542	1	K00827.01	FP	0.673	-	-	-	0
11922778	1	K03408.01	CAND	0.671	-	-	-	0
3229150	1	K02150.01	CAND	0.671	-	-	-	0
9898447	1	K02803.01	CAND	0.668	0.832	-	-	0
8261920	2	K02174.02	CAND	0.668	0.549	-	-	0
8984831	4	-	-	0.667	-	-	-	0
6721523	5	-	-	0.667	0.462	-	-	0
5480766	5	-	-	0.666	-	-	-	0
10074466	1	K01917.01	CAND	0.663	0.606	-	-	0
5980783	1	K02967.01	CAND	0.663	0.662	0.854	0.968	0
9700145	4	-	-	0.661	0.474	-	-	0
5629353	3	K06132.03	CAND	0.66	0.791	-	-	1
8162789	1	K00521.01	CAND	0.659	-	-	0.457	0
4547480	2	-	-	0.659	0.53	-	-	1
7202957	1	K02687.01	CAND	0.657	-	-	-	0
6862721	1	K01982.01	CAND	0.657	0.746	-	-	0
8630788	3	K01258.03	CAND	0.657	0.494	-	-	0
11818607	1	K02467.01	CAND	0.656	0.567	-	-	0
3757778	2	-	-	0.655	0.805	0.933	-	1
4650733	1	K01659.01	CAND	0.655	0.585	0.747	0.913	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

10158418	1	K01784.01	CAND	0.655	0.699	-	-	1
12061969	2	K02061.02	CAND	0.654	-	-	-	0
9528430	2	-	-	0.654	-	-	-	1
4814502	1	K04004.01	CAND	0.653	0.869	1.0	0.969	0
9346253	2	-	-	0.653	0.828	0.995	0.83	1
11968463	3	K02433.03	CAND	0.653	-	0.997	-	0
3561464	3	K03398.03	CAND	0.652	0.494	-	-	0
9307509	1	K02096.01	CAND	0.652	0.857	-	-	0
8410415	1	K02291.01	CAND	0.651	0.773	-	-	0
6805146	2	-	-	0.651	0.736	0.869	0.79	1
7700622	1	K00315.01	CAND	0.651	0.708	-	-	0
3937519	2	K00221.02	CAND	0.649	0.824	-	-	0
10397751	3	K02859.03	CAND	0.648	-	-	-	0
10621643	5	-	-	0.647	0.396	-	-	0
10141900	1	K01082.01	CAND	0.645	0.694	-	-	0
10252275	1	K03130.01	CAND	0.644	0.8	-	-	0
9458754	4	-	-	0.642	0.411	-	-	0
8394756	3	-	-	0.642	-	-	-	0
2447832	5	-	-	0.642	0.421	-	-	0
11122789	4	-	-	0.64	-	-	-	1
3561464	2	K03398.02	CAND	0.639	0.669	-	-	0
11967788	2	K04021.02	CAND	0.639	0.585	-	-	0
4846856	1	K02900.01	CAND	0.638	-	-	-	0
9995771	1	K03470.01	CAND	0.636	0.821	1.0	0.957	0
10189546	1	K00427.01	CAND	0.636	-	-	-	0
7289577	1	K01974.01	CAND	0.635	0.566	-	-	0
11076400	1	K02759.01	CAND	0.634	0.535	-	-	0
12115188	2	-	-	0.633	0.792	1.0	0.462	0
12406807	1	K03091.01	CAND	0.633	0.601	-	-	0
9100953	2	K04500.02	CAND	0.632	-	-	-	0
11122894	3	K01426.03	CAND	0.632	0.831	-	-	0
5966810	1	K03909.01	CAND	0.631	0.599	0.528	0.87	0
7219825	3	K00238.03	CAND	0.631	0.58	-	-	0
7661065	1	K02307.01	CAND	0.63	0.571	-	-	0
10464078	2	-	-	0.63	0.551	1.0	0.871	1
8747865	5	-	-	0.63	-	-	-	0
6955650	1	-	-	0.629	0.518	-	-	0
7377343	1	K05384.01	CAND	0.629	0.813	-	-	1
2832589	1	K01942.01	CAND	0.628	-	-	-	0
11601584	3	K01831.03	CAND	0.627	-	-	-	1
5621333	1	K03341.01	CAND	0.627	0.646	-	-	0
9475552	2	K02694.02	CAND	0.627	0.711	-	-	0
11189127	2	-	-	0.625	0.768	0.948	0.864	1
8074328	1	K02486.01	CAND	0.623	-	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

6356692	1	K02948.01	CAND	0.622	0.809	-	-	0
7124026	1	K02275.01	FP	0.621	0.791	-	-	0
8247770	1	K02569.01	CAND	0.621	0.703	-	-	0
12268190	7	-	-	0.621	-	-	-	0
8580438	2	-	-	0.62	0.705	0.846	-	1
10865397	4	-	-	0.62	-	-	-	0
5080636	2	K01843.02	CAND	0.619	-	-	-	0
5905822	1	K02801.01	CAND	0.617	0.682	-	-	0
9958962	3	K00593.03	CAND	0.617	-	-	-	0
9654468	1	K02417.01	CAND	0.615	0.615	-	-	0
1571511	2	-	-	0.614	0.666	0.999	0.979	1
10470206	1	K00335.01	FP	0.613	-	-	-	0
10186945	2	-	-	0.612	0.657	0.837	0.823	1
3634051	1	K06103.01	FP	0.612	0.587	-	-	0
6115603	3	-	-	0.612	0.403	-	-	0
6605493	1	K02559.01	CAND	0.612	0.731	-	-	0
9140402	1	K00928.01	CAND	0.612	0.935	-	-	1
2164169	1	K01029.01	CAND	0.612	0.744	-	-	0
4947556	1	K03936.01	FP	0.611	-	-	-	0
3629967	1	K01901.01	CAND	0.61	0.614	-	-	0
8321314	1	K02293.01	CAND	0.61	0.699	-	-	0
7678434	2	K00892.02	CAND	0.61	0.522	-	-	1
10525049	1	K04252.01	CAND	0.61	0.806	-	-	0
10656508	1	K00211.01	CAND	0.61	-	-	-	1
9153823	1	K06195.01	FP	0.608	0.817	-	-	0
3120650	5	-	-	0.608	0.38	-	-	0
10028140	1	K01591.01	CAND	0.608	0.636	-	-	0
6428794	1	K04054.01	CAND	0.607	0.847	0.999	0.914	0
5441980	1	K00607.01	CAND	0.606	0.802	-	-	0
9472000	1	K02082.01	CAND	0.606	0.541	-	-	0
8492101	1	K04437.01	FP	0.606	-	-	-	0
8013439	3	K02352.03	CAND	0.604	0.778	-	-	1
8397947	1	K04772.01	CAND	0.604	-	-	-	0
12737015	1	K04245.01	CAND	0.604	-	-	-	0
2161536	1	K02130.01	CAND	0.603	0.674	-	-	0
10744342	4	-	-	0.603	0.558	-	-	0
7879433	1	K02527.01	CAND	0.603	0.632	-	-	0
7987866	1	K02363.01	CAND	0.601	0.728	-	-	0
8261920	1	K02174.01	CAND	0.601	0.503	-	-	0
5131180	1	K00641.01	CAND	0.601	0.641	-	-	0
7050754	2	-	-	0.601	-	-	-	0
1872821	1	K02351.01	CAND	0.599	0.579	-	-	0
4770174	1	K02971.01	CAND	0.598	0.747	-	-	0
8709688	1	K03019.01	CAND	0.598	0.709	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

3323887	3	-	-	0.598	0.42	0.999	-	0
7120108	1	K02325.01	CAND	0.597	0.602	-	-	0
2718778	1	K02489.01	CAND	0.596	-	0.998	0.857	0
8189801	1	K02480.01	CAND	0.596	0.655	-	-	0
10460984	3	K00474.03	CAND	0.595	-	-	-	0
8823868	2	-	-	0.595	0.705	0.835	0.406	1
9017682	1	K01630.01	CAND	0.593	0.648	-	-	0
6359320	3	K01127.03	CAND	0.593	-	-	-	0
9334893	1	K02298.01	CAND	0.593	0.492	-	-	0
8110767	1	K02504.01	CAND	0.592	0.514	-	-	0
9451127	2	-	-	0.592	0.678	-	0.353	1
11818872	1	K02581.01	CAND	0.592	0.803	-	-	0
3218908	4	-	-	0.591	0.626	0.585	0.514	0
10614158	2	-	-	0.591	-	-	-	1
4851530	1	K01884.01	FP	0.591	-	-	-	0
4157325	3	K01860.03	CAND	0.59	-	-	-	0
5213230	1	K03474.01	CAND	0.59	0.72	0.748	0.933	0
1722916	6	-	-	0.59	-	-	-	0
7826659	1	K02686.01	CAND	0.59	0.584	-	-	0
7869917	2	K01525.02	CAND	0.589	-	-	-	0
10092312	1	K01823.01	FP	0.589	-	-	-	0
9283002	3	-	-	0.588	0.431	-	-	0
4771030	3	-	-	0.588	-	-	-	0
5556241	4	-	-	0.587	-	-	-	0
5942808	2	K02250.02	CAND	0.585	-	-	-	0
4385148	2	K02942.02	CAND	0.585	0.498	-	-	0
7463685	1	K02890.01	CAND	0.583	0.674	-	-	0
3103212	1	K04145.01	CAND	0.583	0.841	-	-	0
10028352	1	K01957.01	CAND	0.583	0.74	-	-	1
6871071	4	K02220.04	CAND	0.582	0.589	-	-	0
5942808	1	K02250.01	CAND	0.581	-	-	-	0
7199397	1	K00075.01	CAND	0.581	-	-	-	0
12457978	6	-	-	0.581	-	-	-	0
6867588	1	K02571.01	CAND	0.58	0.805	-	-	0
4173026	1	K02172.01	CAND	0.579	0.778	-	-	0
4473226	1	K02229.01	CAND	0.579	0.802	-	-	0
7375348	2	K00266.02	CAND	0.579	0.645	0.993	0.846	1
6110119	8	-	-	0.578	-	-	-	0
12406749	1	K01476.01	CAND	0.577	0.923	-	-	0
3122913	1	K02490.01	CAND	0.576	0.636	-	-	0
4548011	2	K04288.02	CAND	0.575	0.601	-	-	0
5436013	1	K02471.01	CAND	0.575	0.872	-	-	0
8753896	2	-	-	0.574	0.528	0.555	-	0
9714123	2	-	-	0.573	0.755	0.966	0.429	1

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

11395587	1	K00350.01	CAND	0.573	0.907	-	-	0
9691311	4	-	-	0.572	-	-	-	0
10533616	3	-	-	0.571	-	-	-	0
9936080	1	K04431.01	CAND	0.57	0.554	-	-	0
9205938	2	K02162.02	CAND	0.57	-	-	-	0
7017372	1	K03689.01	CAND	0.569	-	-	-	0
10583180	2	-	-	0.569	-	-	-	1
7017274	1	K03209.01	CAND	0.569	0.641	-	-	0
10471621	1	K02554.01	CAND	0.568	0.499	-	-	0
9003401	1	K02319.01	CAND	0.567	-	-	-	0
9714572	8	-	-	0.567	-	-	-	0
9328641	2	-	-	0.567	-	-	-	0
6837283	1	K02914.01	CAND	0.567	-	-	-	0
9002538	2	K03196.02	CAND	0.566	-	-	-	0
9993529	1	K02644.01	FP	0.565	-	-	-	0
11968463	4	K02433.04	CAND	0.565	0.49	-	-	0
2444412	1	K00103.01	CAND	0.565	0.459	0.924	-	0
7137725	1	K01699.01	FP	0.564	0.574	-	-	0
3231341	5	-	-	0.564	0.555	0.894	0.877	0
8480582	1	K04386.01	CAND	0.561	0.91	-	-	0
5621333	2	K03341.02	CAND	0.561	0.557	-	-	0
12170648	2	-	-	0.56	-	0.783	0.429	1
10122538	5	K02926.05	CAND	0.559	0.673	-	-	0
6579806	2	-	-	0.558	-	-	-	1
10875007	2	K04149.02	CAND	0.558	-	-	-	0
8056665	2	K00089.02	CAND	0.558	0.55	-	-	0
7256914	2	K04136.02	CAND	0.557	0.678	-	-	0
5906694	1	K04069.01	CAND	0.557	0.556	-	-	0
3239671	1	K02066.01	CAND	0.557	0.515	-	-	0
8540376	1	K07892.01	CAND	0.557	0.577	-	-	0
4058206	3	-	-	0.555	-	-	-	0
11394027	1	K00349.01	CAND	0.554	0.931	-	-	0
8240861	4	-	-	0.553	-	-	-	1
8554498	1	K00005.01	CAND	0.553	-	-	-	1
7825899	3	K00896.03	CAND	0.553	-	-	-	0
5471059	2	K03956.02	FP	0.553	-	-	-	0
11548140	1	K00256.01	FP	0.552	0.658	-	-	1
7749773	1	K02848.01	CAND	0.552	0.634	-	-	0
8684730	1	K00319.01	CAND	0.551	0.575	-	-	0
9872292	1	K01537.01	CAND	0.551	0.704	-	-	0
6606438	1	K02860.01	CAND	0.551	-	-	-	0
12602335	3	-	-	0.551	-	-	-	0
5775129	1	K02802.01	CAND	0.551	0.719	0.972	0.947	0
10005758	2	K01783.02	CAND	0.55	0.548	0.993	0.737	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

5881120	1	K04156.01	CAND	0.55	0.621	-	-	0
5552761	7	-	-	0.55	-	-	-	0
4140813	1	K02117.01	CAND	0.549	0.779	-	-	0
5773121	1	K04002.01	CAND	0.549	0.652	0.658	0.86	0
9071593	2	K02257.02	CAND	0.549	0.573	-	-	0
4830605	1	K01660.01	CAND	0.548	-	-	-	0
7841925	2	K01499.02	CAND	0.548	-	-	-	0
9283156	1	K02657.01	CAND	0.547	0.625	-	-	0
11700604	7	-	-	0.547	-	-	-	0
7287118	4	-	-	0.547	-	-	-	0
3967219	3	-	-	0.546	-	-	-	0
9661979	1	K02132.01	CAND	0.546	0.673	-	-	0
12217824	3	-	-	0.546	-	-	-	0
4665571	1	K02393.01	CAND	0.545	-	-	-	0
7102227	3	K01360.03	CAND	0.544	0.708	-	-	0
10397751	4	K02859.04	CAND	0.544	-	-	-	0
10924400	2	-	-	0.544	0.757	-	-	0
12164634	3	-	-	0.544	0.347	-	0.556	1
9086251	1	K02367.01	CAND	0.543	0.547	-	-	0
5450893	1	K02970.01	CAND	0.543	0.7	-	-	0
6268648	3	K01613.03	CAND	0.543	0.585	-	-	1
7582608	4	-	-	0.543	-	-	-	0
8880123	1	K03493.01	CAND	0.541	0.641	-	-	0
7032429	5	-	-	0.54	-	-	-	0
6677841	3	-	-	0.54	0.531	0.829	0.571	0
6300348	1	K00212.01	CAND	0.54	0.899	-	-	0
6707942	2	-	-	0.54	0.682	0.987	-	1
5360082	2	-	-	0.539	0.854	-	-	0
5357545	1	K02219.01	CAND	0.539	-	-	-	0
5544450	1	K03226.01	CAND	0.538	0.675	-	-	0
6632435	2	-	-	0.538	-	-	-	0
11513486	1	K04748.01	CAND	0.538	-	-	-	0
10351231	1	K01166.01	CAND	0.537	0.656	-	-	0
9009953	1	K04014.01	CAND	0.536	-	-	-	0
6441738	1	K01246.01	CAND	0.536	0.67	-	-	0
6889235	2	-	-	0.536	-	-	-	1
8651389	1	K01754.01	CAND	0.536	0.679	0.984	0.948	0
5092266	1	K03045.01	CAND	0.535	0.627	0.521	0.98	0
10922659	1	K03335.01	CAND	0.534	0.719	-	-	0
6775034	5	-	-	0.534	-	-	-	1
8013419	1	K00901.01	CAND	0.534	0.566	-	-	0
8260218	1	K01066.01	CAND	0.533	0.475	-	-	0
7100673	5	K04032.05	CAND	0.532	-	-	-	0
10813078	1	K05831.01	CAND	0.532	0.575	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

9117416	1	K03425.01	CAND	0.532	-	-	-	1
5008245	1	K03770.01	CAND	0.531	-	-	-	0
6268648	4	-	-	0.531	-	-	-	1
4945266	2	-	-	0.531	-	-	-	0
9491205	3	-	-	0.531	-	-	0.385	0
9092504	2	K02716.02	CAND	0.53	-	-	-	0
8874090	2	K01404.02	CAND	0.53	-	-	-	0
7035274	3	-	-	0.53	-	-	-	0
11601584	4	K01831.04	CAND	0.529	-	0.999	0.448	1
7939330	1	K01581.01	CAND	0.527	0.532	-	-	0
4912650	1	K05098.01	CAND	0.526	0.516	0.66	-	0
7363829	1	-	-	0.525	-	0.553	-	0
10345478	2	-	-	0.524	0.632	-	-	1
5724810	1	-	-	0.524	0.526	-	-	0
8935810	1	K01395.01	CAND	0.523	0.808	-	-	0
8707670	5	-	-	0.523	-	-	-	0
4385148	1	K02942.01	CAND	0.521	0.668	-	-	0
9166700	1	K05632.01	CAND	0.521	0.591	-	-	0
8265218	1	K00522.01	CAND	0.52	0.735	-	-	0
6636320	1	K02989.01	CAND	0.52	-	-	-	0
7768451	1	K01527.01	CAND	0.52	0.67	-	-	0
5613821	1	K02915.01	CAND	0.518	0.795	-	-	0
8524712	3	-	-	0.517	-	-	-	0
12069786	3	-	-	0.517	-	-	-	0
6183511	1	K02542.01	CAND	0.516	0.635	-	-	0
9758089	2	K01871.02	CAND	0.516	-	-	-	1
8456679	1	K00102.01	CAND	0.514	-	-	-	0
8702921	3	-	-	0.513	0.405	-	-	1
4840513	2	-	-	0.512	0.946	1.0	0.922	0
8008067	3	K00316.03	CAND	0.512	-	-	-	0
5124667	1	K01822.01	CAND	0.512	0.736	0.722	0.82	0
8240861	3	-	-	0.512	-	-	-	1
8760040	2	K02963.02	CAND	0.512	-	-	-	0
9490506	2	-	-	0.511	-	-	-	0
5372081	7	-	-	0.51	-	-	-	0
12164634	6	-	-	0.509	-	-	-	1
7533425	6	-	-	0.509	-	-	-	0
8758204	1	K02841.01	CAND	0.509	0.547	-	-	0
7869590	2	-	-	0.509	-	-	-	0
7872212	7	-	-	0.507	-	-	-	0
3112129	2	-	-	0.507	-	0.977	0.8	0
9279865	3	-	-	0.507	-	-	-	0
12009347	2	-	-	0.506	0.738	0.999	0.86	0
10187017	6	K00082.06	CAND	0.506	-	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

4939265	3	-	-	0.506	0.659	-	0.462	0
9673338	9	-	-	0.505	-	-	-	0
7117284	2	-	-	0.504	0.44	-	-	0
3122872	1	K03203.01	CAND	0.503	0.769	-	-	0
5963582	1	K02479.01	CAND	0.503	0.668	-	-	0
11046025	1	K01646.01	CAND	0.502	-	-	-	0
11450414	3	K01992.03	FP	0.502	-	-	-	0
11177543	1	K01648.01	CAND	0.501	0.605	-	-	0
4370956	2	-	-	0.501	-	-	-	0
9412462	3	-	-	0.5	-	-	-	1
10853783	2	-	-	0.5	-	-	-	0
11465651	1	K04150.01	CAND	0.499	-	-	-	0
9579208	1	K03165.01	CAND	0.499	-	-	-	0
5792202	4	K00841.04	FP	0.497	0.602	-	-	0
9700145	2	-	-	0.497	-	-	-	0
10798331	1	K02373.01	CAND	0.496	-	-	-	0
4077526	4	K01336.04	CAND	0.496	-	-	-	0
3834322	2	-	-	0.496	0.562	0.525	0.375	1
5004909	1	K03421.01	FP	0.496	0.516	-	-	0
3453026	2	-	-	0.496	-	-	-	0
7115249	7	-	-	0.494	-	-	-	0
8414216	1	K07036.01	FP	0.494	0.766	-	-	0
6606934	2	-	-	0.493	-	0.508	-	1
9086154	1	K04060.01	CAND	0.491	0.481	-	-	0
9588822	2	K04388.02	CAND	0.49	0.672	-	-	1
12257999	1	K02577.01	CAND	0.49	0.539	-	-	0
12316447	3	-	-	0.489	-	-	-	1
8241079	1	K03020.01	CAND	0.488	0.642	-	-	1
7031126	7	-	-	0.487	-	-	-	0
6432345	1	K02757.01	CAND	0.486	0.705	0.999	0.943	0
6115603	10	-	-	0.486	-	-	-	0
10717241	2	K00430.02	CAND	0.486	0.495	-	-	0
5372966	2	-	-	0.485	0.812	0.977	0.99	1
2443393	1	K02603.01	CAND	0.485	-	-	-	0
10482774	7	-	-	0.485	-	-	-	0
6945500	5	-	-	0.484	-	-	-	0
5095635	1	K02607.01	CAND	0.483	0.529	0.995	0.823	0
7031126	3	-	-	0.483	-	-	-	0
10514770	2	-	-	0.481	-	-	0.458	1
2993589	5	-	-	0.481	-	-	-	0
10659313	2	-	-	0.48	-	-	-	1
8690001	7	-	-	0.479	-	-	-	1
8280511	5	K01151.05	CAND	0.479	-	-	-	0
9512981	1	K01466.01	CAND	0.479	-	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

7352016	1	-	-	0.478	-	-	-	0
7047496	2	-	-	0.478	-	-	-	0
8176653	1	K01016.01	FP	0.477	-	-	-	1
7673192	5	K02722.05	CAND	0.477	-	-	-	0
7032218	3	-	-	0.477	-	-	-	0
4566740	8	-	-	0.477	-	-	-	0
4566740	9	-	-	0.476	-	-	-	0
11127641	5	-	-	0.474	-	-	-	0
10332789	2	-	-	0.471	0.726	0.923	-	1
8280511	4	K01151.04	CAND	0.471	-	-	-	0
6756202	3	-	-	0.471	-	-	-	1
3561700	5	-	-	0.471	-	-	-	0
2438502	2	-	-	0.471	0.786	-	0.481	0
3660392	6	-	-	0.471	-	-	-	0
5302881	2	-	-	0.47	-	-	-	0
8702921	2	-	-	0.47	-	-	-	1
7211759	8	-	-	0.468	-	-	-	0
10028792	3	K01574.03	CAND	0.466	-	-	-	0
5476864	2	-	-	0.464	0.529	-	-	0
6140084	10	-	-	0.464	-	-	-	0
7117444	6	-	-	0.464	-	-	-	0
10152836	2	-	-	0.463	0.555	0.535	-	1
8826317	10	-	-	0.462	-	-	-	0
10206675	2	-	-	0.462	-	-	-	0
4263801	4	-	-	0.462	-	-	-	0
9099950	7	-	-	0.462	-	-	-	0
11100532	9	-	-	0.461	-	-	-	0
8242681	2	-	-	0.461	-	-	0.455	1
8527297	3	-	-	0.461	-	-	-	0
9032900	4	-	-	0.459	0.446	-	-	1
1433399	3	-	-	0.458	-	-	-	0
9237305	5	-	-	0.458	-	-	-	0
7041634	2	-	-	0.458	0.514	-	-	0
3450040	1	K04205.01	CAND	0.458	0.775	-	-	0
10397751	5	K02859.05	CAND	0.458	-	-	-	0
3975085	7	-	-	0.458	-	-	-	0
10132618	2	-	-	0.455	-	-	-	0
8971294	2	-	-	0.455	-	-	-	0
7335514	1	K06860.01	CAND	0.455	-	-	-	0
3862246	2	-	-	0.455	0.565	-	0.429	1
5787972	10	-	-	0.454	-	-	-	0
10877367	6	-	-	0.454	-	-	-	0
5088084	4	-	-	0.454	-	-	-	0
10552700	4	-	-	0.452	-	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

9457728	1	K04637.01	CAND	0.449	0.646	-	0.429	0
2712825	1	K08086.01	FP	0.449	-	-	-	0
6783562	8	-	-	0.446	-	-	-	0
9642018	2	-	-	0.445	-	-	-	1
3120650	4	-	-	0.445	-	-	-	0
8950019	8	-	-	0.444	-	-	-	0
7880676	2	-	-	0.442	0.473	-	-	0
9278696	3	-	-	0.441	-	-	-	0
8783270	5	-	-	0.44	-	-	-	0
2436450	6	-	-	0.439	-	-	-	0
8553462	10	-	-	0.439	-	-	-	0
8431611	3	-	-	0.438	-	-	-	0
4739229	3	-	-	0.436	-	-	-	0
8176564	1	K02720.01	CAND	0.435	-	-	-	0
6945362	8	-	-	0.434	-	-	-	0
9178894	3	-	-	0.433	-	-	-	0
1026133	2	-	-	0.433	-	-	-	0
12644769	2	-	-	0.432	-	-	-	1
6041680	3	-	-	0.43	-	-	-	0
9156461	10	-	-	0.43	-	-	-	0
7533425	5	-	-	0.43	-	-	-	0
8894646	1	K02641.01	CAND	0.429	0.485	-	-	0
10843431	2	K07378.02	CAND	0.428	-	-	-	0
9006186	4	K02169.04	CAND	0.424	-	-	-	0
9149789	2	-	-	0.423	-	0.393	-	1
8096395	1	K07587.01	CAND	0.423	-	-	-	0
6387450	2	-	-	0.423	0.43	-	-	1
10652379	10	-	-	0.423	-	-	-	0
3764879	1	K02141.01	FP	0.423	-	-	-	0
7767559	1	K00895.01	FP	0.422	-	-	-	1
11356602	1	K07439.01	FP	0.42	-	-	-	0
4347340	4	-	-	0.42	-	-	-	0
10395312	10	-	-	0.413	-	-	-	0
3247616	4	-	-	0.413	-	-	-	0
8631160	1	K01271.01	CAND	0.411	-	-	-	1
11968463	6	-	-	0.411	-	-	-	0
7446357	10	-	-	0.41	-	-	-	0
2436450	7	-	-	0.409	-	-	-	0
8328003	6	-	-	0.408	0.41	-	-	0
5812648	4	-	-	0.408	-	-	-	0
8840117	8	-	-	0.408	-	-	-	0
6762829	5	-	-	0.406	0.409	0.893	0.5	1
7446357	8	-	-	0.402	-	-	-	0
6721523	7	-	-	0.401	-	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

6500206	1	K02451.01	FP	0.401	0.418	-	-	0
8362880	2	-	-	0.399	-	-	-	0
7900137	2	-	-	0.398	-	-	-	0
5471059	1	K03956.01	FP	0.397	-	-	-	0
7816992	10	-	-	0.395	-	-	-	0
5265699	9	-	-	0.394	-	-	-	0
9574614	1	K01745.01	FP	0.391	-	-	-	1
1718958	1	K04053.01	CAND	0.391	0.411	-	-	0
9084778	1	K01631.01	FP	0.39	-	-	-	1
5295670	4	-	-	0.387	-	-	-	0
6277594	2	-	-	0.386	-	-	-	0
8396405	1	K01733.01	CAND	0.383	0.517	-	-	0
4157325	4	K01860.04	FP	0.382	0.592	-	-	0
8613535	4	-	-	0.381	-	-	-	0
9166826	7	-	-	0.381	-	-	-	0
11919968	9	-	-	0.379	-	-	-	0
8818497	2	-	-	0.373	-	-	-	0
8703129	1	K02758.01	CAND	0.37	-	-	-	1
3558849	1	K04307.01	CAND	0.364	-	-	-	0
4066898	3	-	-	0.362	-	-	-	0
10801647	8	-	-	0.356	-	-	-	0
10321305	8	-	-	0.354	-	-	-	0
7115878	5	-	-	0.353	-	-	-	0
5652983	1	K00371.01	FP	0.352	-	-	-	1
10031808	6	-	-	0.351	-	-	-	1
8460081	5	-	-	0.35	-	-	-	0
4145040	4	-	-	0.347	-	-	-	0
7582608	6	-	-	0.345	-	-	-	0
4735826	2	-	-	-	0.981	-	-	0
8953059	2	-	-	-	0.922	0.999	0.879	1
8822421	1	K06188.01	FP	-	0.888	-	-	0
7097965	1	K02083.01	CAND	-	0.881	-	-	0
11251058	1	K07429.01	FP	-	0.847	-	-	0
9896435	2	-	-	-	0.84	-	-	1
8142942	1	K01985.01	CAND	-	0.83	-	-	0
12647577	1	K04530.01	CAND	-	0.827	-	-	0
3097352	2	-	-	-	0.813	0.513	-	1
9427402	1	K01397.01	CAND	-	0.812	-	-	0
1026957	1	K00958.01	CAND	-	0.808	0.788	0.866	1
11656302	2	-	-	-	0.803	-	-	1
8742590	1	K01281.01	CAND	-	0.792	-	-	0
4458082	2	K02303.02	CAND	-	0.791	-	-	0
10015937	1	K01720.01	CAND	-	0.777	-	-	0
7051180	1	K00064.01	CAND	-	0.771	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

11904734	1	K01764.01	CAND	-	0.768	-	-	0
7779077	1	K01842.01	CAND	-	0.767	-	-	0
10354039	1	K01159.01	CAND	-	0.763	-	-	0
10917681	1	K01963.01	CAND	-	0.763	-	-	0
3648437	1	K01861.01	CAND	-	0.756	-	-	0
10094670	1	K01984.01	CAND	-	0.748	-	-	0
8256453	1	K02573.01	CAND	-	0.747	-	-	0
6291837	1	K00308.01	CAND	-	0.746	-	-	0
5003117	2	-	-	-	0.742	-	0.556	1
10843431	1	K07378.01	CAND	-	0.741	0.961	-	0
8949925	1	K02972.01	CAND	-	0.733	-	-	0
7103919	1	K04310.01	CAND	-	0.732	-	-	0
3967326	1	K02084.01	CAND	-	0.729	-	-	0
9528420	2	-	-	-	0.721	0.995	0.657	1
7017274	2	K03209.02	CAND	-	0.718	-	-	0
10736489	1	K07368.01	CAND	-	0.717	-	-	0
12017140	2	-	-	-	0.702	0.512	-	1
7021681	2	K00255.02	CAND	-	0.701	-	-	0
8581240	1	K03111.01	CAND	-	0.699	-	-	0
4665571	2	K02393.02	CAND	-	0.699	0.681	0.821	0
9758089	1	K01871.01	CAND	-	0.696	-	-	1
6185476	1	-	-	-	0.69	-	-	0
5816165	1	K01043.01	FP	-	0.688	0.436	0.471	0
3756264	1	K03108.01	CAND	-	0.687	-	-	0
12456063	1	K04329.01	CAND	-	0.685	-	-	0
8605074	2	-	-	-	0.683	0.902	-	1
8230809	2	-	-	-	0.682	-	-	1
8591693	1	K02123.01	CAND	-	0.679	-	-	0
7750419	1	K01708.01	CAND	-	0.679	-	-	0
6707908	1	K02014.01	CAND	-	0.678	-	-	0
8158429	1	K05482.01	CAND	-	0.677	-	-	0
9602613	1	K02612.01	CAND	-	0.675	-	-	0
9635606	1	K02535.01	CAND	-	0.67	-	-	0
9715925	2	-	-	-	0.669	-	-	1
4912991	2	-	-	-	0.669	-	0.423	1
4069213	2	-	-	-	0.664	0.687	-	1
8845026	2	-	-	-	0.663	-	-	1
9834731	1	K03043.01	CAND	-	0.661	-	-	0
2719873	2	-	-	-	0.651	-	-	1
5705819	1	K02091.01	CAND	-	0.649	-	-	0
8617363	1	K02945.01	CAND	-	0.648	-	-	0
9886224	2	-	-	-	0.641	0.988	-	1
3554031	2	K01194.02	FP	-	0.635	-	-	0
10155029	1	K03208.01	CAND	-	0.632	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

11013201	2	K00972.02	FP	-	0.631	-	-	1
5475042	1	K03050.01	CAND	-	0.63	-	-	0
11856178	1	K06246.01	CAND	-	0.629	-	-	0
7605600	2	-	-	-	0.629	0.997	-	1
4676964	1	K03069.01	CAND	-	0.627	-	-	0
9472074	1	K02735.01	CAND	-	0.625	-	-	0
3545135	1	K02755.01	CAND	-	0.62	-	-	0
5003670	1	K04524.01	CAND	-	0.619	-	-	0
4919550	1	K04766.01	CAND	-	0.617	-	-	0
8824737	1	K03347.01	CAND	-	0.615	-	-	0
5098444	2	-	-	-	0.615	-	-	1
2975770	2	K01788.02	FP	-	0.614	-	-	0
8022489	1	K02674.01	CAND	-	0.614	-	-	1
5308537	1	K04409.01	CAND	-	0.608	-	-	0
1724719	2	K04212.02	CAND	-	0.606	-	-	0
6523351	1	K03117.01	CAND	-	0.606	-	-	0
12208631	1	K02449.01	CAND	-	0.602	-	-	0
4150390	2	-	-	-	0.596	-	-	1
9011877	1	K05597.01	CAND	-	0.595	-	-	0
12602314	1	K02853.01	CAND	-	0.595	-	-	0
7609674	1	K03128.01	CAND	-	0.594	-	-	0
5979863	2	-	-	-	0.593	-	0.667	1
11124436	1	K04442.01	CAND	-	0.586	-	-	0
5270698	2	-	-	-	0.578	0.999	-	0
7106173	1	K03083.01	CAND	-	0.575	-	-	0
9411317	2	-	-	-	0.574	-	-	1
7446631	1	K02598.01	CAND	-	0.572	-	-	0
5003670	2	K04524.02	CAND	-	0.569	-	-	0
10514430	4	-	-	-	0.568	-	-	0
11200405	2	-	-	-	0.567	-	-	1
4857058	1	K03061.01	CAND	-	0.565	-	-	0
11081504	1	K04287.01	CAND	-	0.565	-	-	0
2973386	1	K03034.01	CAND	-	0.565	-	-	0
5211199	2	K02158.02	CAND	-	0.563	-	-	0
9957627	1	K00592.01	CAND	-	0.563	-	-	0
10141900	2	K01082.02	CAND	-	0.558	-	-	0
10597693	1	K02958.01	CAND	-	0.555	-	-	0
7918478	1	K04209.01	CAND	-	0.553	-	-	0
3559860	1	K03440.01	FP	-	0.551	-	-	0
9836563	1	K04421.01	CAND	-	0.551	-	-	0
9020114	1	K03088.01	CAND	-	0.549	-	-	0
5428657	1	K06130.01	CAND	-	0.549	-	-	0
10874215	1	K02305.01	CAND	-	0.548	-	-	0
10265898	2	K00732.02	CAND	-	0.546	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

8827575	1	K03052.01	CAND	-	0.544	-	-	0
11125797	1	K03371.01	CAND	-	0.543	-	-	0
7512982	2	K01480.02	CAND	-	0.543	-	-	0
6131236	1	K01051.01	CAND	-	0.542	-	-	0
8804283	2	K01276.02	CAND	-	0.537	-	-	0
4172805	1	K04427.01	CAND	-	0.536	-	-	0
5106313	1	K02878.01	CAND	-	0.536	-	-	0
12061969	1	K02061.01	FP	-	0.536	-	-	0
8390826	1	-	-	-	0.535	-	-	0
6041734	2	-	-	-	0.534	0.919	0.87	1
5450814	2	-	-	-	0.531	-	-	1
4860678	2	K01602.02	CAND	-	0.53	-	-	0
9552608	2	-	-	-	0.527	-	0.458	1
7107802	2	K02420.02	CAND	-	0.527	-	-	0
9002538	1	K03196.01	CAND	-	0.527	-	-	0
5897826	3	-	-	-	0.527	-	-	1
5544450	2	K03226.02	CAND	-	0.526	-	-	0
6945786	1	K03136.01	CAND	-	0.526	-	-	0
9716028	1	K04273.01	CAND	-	0.525	-	-	0
6061773	1	K02001.01	CAND	-	0.525	-	-	0
2574338	1	K01030.01	CAND	-	0.525	0.765	0.949	0
9474222	2	-	-	-	0.524	0.998	-	1
11599038	2	K01437.02	FP	-	0.521	-	-	0
8624520	1	K01816.01	CAND	-	0.521	-	-	1
5103942	2	-	-	-	0.521	-	-	0
2019477	1	K06093.01	FP	-	0.52	-	-	0
4061149	1	K01201.01	CAND	-	0.52	-	-	0
5436338	1	K02835.01	CAND	-	0.519	-	-	0
4060229	1	K04570.01	CAND	-	0.515	-	-	0
9288237	1	K03114.01	CAND	-	0.514	-	-	0
9896018	3	K02579.03	CAND	-	0.513	-	-	0
6026737	1	K02949.01	CAND	-	0.511	-	-	0
11246161	2	-	-	-	0.51	0.878	0.5	0
6593150	1	K03037.01	CAND	-	0.51	0.51	-	0
11465950	1	K02620.01	CAND	-	0.509	0.834	0.959	0
6153407	1	K03179.01	CAND	-	0.508	-	-	0
8838950	1	K02421.01	CAND	-	0.507	-	-	0
11869052	1	K00120.01	CAND	-	0.506	-	-	1
8963721	1	K02233.01	FP	-	0.506	-	-	0
5351250	5	K00408.05	FP	-	0.503	-	-	0
5542466	2	K01590.02	CAND	-	0.502	-	-	0
11517719	2	-	-	-	0.501	-	-	1
9650989	1	K03095.01	CAND	-	0.501	-	-	0
2986833	1	K04875.01	CAND	-	0.501	-	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

10666592	2	-	-	-	0.498	-	-	0
4547603	1	K02855.01	CAND	-	0.497	-	-	0
5906426	1	K02377.01	CAND	-	0.496	-	-	0
7288444	1	K03482.01	CAND	-	0.496	-	-	0
7377033	1	K00882.01	FP	-	0.494	-	-	1
7450747	1	K04267.01	CAND	-	0.494	0.883	-	0
6792908	1	K04341.01	CAND	-	0.494	-	-	0
6268648	2	K01613.02	CAND	-	0.494	-	-	1
1849702	1	K02538.01	CAND	-	0.493	-	-	0
8409588	1	K00690.01	FP	-	0.49	-	-	1
9032900	2	-	-	-	0.481	-	-	1
4770365	3	K01475.03	CAND	-	0.48	-	-	0
12301181	2	K02059.02	CAND	-	0.476	-	-	1
2852941	1	K04298.01	CAND	-	0.476	-	-	0
7222086	1	K01701.01	CAND	-	0.469	-	-	0
12164634	2	-	-	-	0.466	-	-	1
11246161	1	K02796.01	CAND	-	0.465	-	-	0
3448132	2	-	-	-	0.462	-	-	0
4739229	2	-	-	-	0.456	-	-	0
8313667	1	K01145.01	CAND	-	0.455	-	-	0
5551672	1	K03119.01	CAND	-	0.452	0.829	-	0
11509792	1	-	-	-	0.436	-	-	0
3338885	2	K01845.02	CAND	-	0.434	-	-	0
3437940	2	-	-	-	0.424	-	-	0
10863286	2	-	-	-	0.423	-	-	1
5446285	3	-	-	-	0.418	0.7	-	0
12302530	3	-	-	-	0.416	-	0.404	0
10018866	2	-	-	-	0.41	-	-	0
8264404	2	-	-	-	0.406	-	-	0
2984632	2	-	-	-	0.405	-	-	0
5689351	6	-	-	-	0.375	-	-	0
8631751	1	K02453.01	CAND	-	0.373	-	-	0
9897838	1	-	-	-	0.372	-	-	0
5384713	5	-	-	-	0.368	-	-	1
4769931	1	K04058.01	FP	-	0.36	-	-	0
6603756	2	-	-	-	-	1.0	-	1
7972785	2	-	-	-	-	0.999	0.969	1
6756669	2	-	-	-	-	0.996	0.924	1
8416523	2	-	-	-	-	0.995	-	0
5774557	2	-	-	-	-	0.995	-	0
8953257	2	-	-	-	-	0.993	0.841	1
8561063	4	-	-	-	-	0.99	-	0
9705459	2	-	-	-	-	0.985	0.839	1
8236479	1	-	-	-	-	0.978	-	0

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

3558981	2	-	-	-	-	0.976	-	1
4049124	2	-	-	-	-	0.976	-	1
9098590	2	-	-	-	-	0.959	-	1
5608566	2	-	-	-	-	0.947	-	1
6500206	2	-	-	-	-	0.939	-	0
12644769	3	-	-	-	-	0.911	-	1
6451605	1	-	-	-	-	0.906	-	0
8415004	4	-	-	-	-	0.876	-	0
6504534	3	-	-	-	-	0.824	-	1
5446285	2	-	-	-	-	0.823	-	0
3323887	4	-	-	-	-	0.817	-	0
6750902	2	-	-	-	-	0.798	-	0
6185476	3	-	-	-	-	0.781	-	0
7270230	2	-	-	-	-	0.773	-	1
7269881	2	-	-	-	-	0.757	-	0
6185476	2	-	-	-	-	0.757	-	0
12010534	2	-	-	-	-	0.749	-	1
7119481	3	K00566.03	CAND	-	-	0.736	-	0
10073672	1	K02764.01	CAND	-	-	0.732	-	0
5728283	2	-	-	-	-	0.726	-	1
9719636	2	-	-	-	-	0.715	-	1
5114623	8	-	-	-	-	0.708	-	0
6060277	2	-	-	-	-	0.683	-	0
10419211	2	-	-	-	-	0.678	-	1
4932348	2	-	-	-	-	0.655	-	1
9413313	1	-	-	-	-	0.637	-	0
6149910	1	K02469.01	CAND	-	-	0.613	-	0
3634755	2	-	-	-	-	0.606	-	0
5810113	2	-	-	-	-	0.596	-	0
11285136	2	-	-	-	-	0.584	-	1
3547091	2	-	-	-	-	0.575	-	1
10657664	2	-	-	-	-	0.561	-	1
8329629	2	-	-	-	-	0.559	-	0
9202151	1	K01393.01	CAND	-	-	0.55	-	0
8292150	2	-	-	-	-	0.537	-	0
10527135	2	-	-	-	-	0.536	-	0
2720354	2	-	-	-	-	0.533	-	1
6462874	2	K01231.02	FP	-	-	0.522	-	0
11391018	2	-	-	-	-	0.5	-	0
9791509	3	-	-	-	-	0.474	-	0
11252617	2	K06236.02	FP	-	-	0.466	-	1
3342467	2	-	-	-	-	0.458	-	0
6209347	5	-	-	-	-	0.414	-	1
7186665	2	-	-	-	-	-	0.75	1

Continued on next page

Table 12: Confirmed planets ordered by LR. Eclipsing Binaries ($eb=1$)

4773392	1	K04367.01	CAND	-	-	-	0.538	0
6762829	4	-	-	-	-	-	0.514	1
7186665	1	K03033.01	FP	-	-	-	0.5	1
5014753	2	-	-	-	-	-	0.5	1
3851949	2	-	-	-	-	-	0.458	1
5357470	2	-	-	-	-	-	0.455	0
3962728	2	-	-	-	-	-	0.452	0
7051984	2	-	-	-	-	-	0.452	1
8695779	2	-	-	-	-	-	0.444	1
6866228	2	-	-	-	-	-	0.444	1
5982353	2	-	-	-	-	-	0.429	1
8038679	1	-	-	-	-	-	0.424	0
6609270	2	K03090.02	FP	-	-	-	0.409	0
9710998	1	-	-	-	-	-	0.406	0
7299905	1	-	-	-	-	-	0.4	0
8681125	1	-	-	-	-	-	0.393	0
6785967	1	-	-	-	-	-	0.379	0
10068030	2	-	-	-	-	-	0.375	1
11249624	3	-	-	-	-	-	0.375	1
10024862	2	-	-	-	-	-	0.375	0
2452440	2	-	-	-	-	-	0.366	1
2163457	1	-	-	-	-	-	0.36	0
12207117	2	-	-	-	-	-	0.359	0
5990753	2	-	-	-	-	-	0.35	1

9.4 Extract of recovered planets ordered by Random Forest (RF_vif) with eclipsing binaries

Table 13 shows the full list of planets. This is an extract of Table 12 but ordered by Random Forest results. Many TCE cases are returned with high probability but they are in systems with eclipsing binaries ($eb=1$) which may explain why they were not included in the KOI table.

Table 13: Confirmed planets ordered by RF_vif Eclipsing Binaries ($eb=1$)

kepid	plnt num	kname	dispos	RF_vif	GB_vif	LR	DT	eb
9838975	2	-	-	0.99	1.0	0.927	0.879	1
3734868	2	-	-	0.99	1.0	0.778	0.643	1
5372966	2	-	-	0.99	0.977	0.485	0.812	1
8552719	2	K01792.02	FP	0.98	1.0	0.949	0.888	1
5092266	1	K03045.01	CAND	0.98	0.521	0.535	0.627	0
5376836	2	-	-	0.98	0.999	0.819	0.869	1
7037540	2	-	-	0.98	0.999	0.956	0.889	1
8180020	2	-	-	0.979	0.999	0.927	0.925	1
9446824	2	-	-	0.979	0.999	0.908	0.913	1
1571511	2	-	-	0.979	0.999	0.614	0.666	1
9411166	2	-	-	0.979	0.974	0.921	0.684	0
8892910	2	-	-	0.978	0.998	0.809	0.591	1
8176650	2	-	-	0.971	1.0	0.905	0.874	0
5370302	2	-	-	0.971	1.0	0.866	0.908	1
3970233	2	-	-	0.97	0.999	0.935	0.972	1
4814502	1	K04004.01	CAND	0.969	1.0	0.653	0.869	0
12504988	2	-	-	0.969	1.0	0.905	0.921	1
7972785	2	-	-	0.969	0.999	-	-	1
5980783	1	K02967.01	CAND	0.968	0.854	0.663	0.662	0
5031857	2	K01573.02	CAND	0.967	-	0.879	0.584	0
6762829	2	-	-	0.96	0.999	0.892	0.839	1
10016874	1	K00426.01	CAND	0.96	0.717	0.85	0.809	0
10904857	2	-	-	0.96	0.994	0.845	0.879	1
7584650	1	K02631.01	CAND	0.96	1.0	0.805	0.885	0
7975727	2	-	-	0.96	0.999	0.804	0.852	1
10616571	3	-	-	0.96	1.0	0.731	0.724	1
11465950	1	K02620.01	CAND	0.959	0.834	-	0.509	0
5978559	1	K02321.01	CAND	0.959	0.992	0.693	0.698	0
10794087	1	K01316.01	CAND	0.959	1.0	0.914	0.899	0
3097926	1	K02186.01	CAND	0.959	0.985	0.837	0.804	0
7023960	2	-	-	0.957	0.996	0.798	0.859	1
9995771	1	K03470.01	CAND	0.957	1.0	0.636	0.821	0
2581316	2	K03681.02	CAND	0.952	1.0	0.942	0.932	0
3342592	2	-	-	0.949	0.998	0.904	0.935	1
5786676	1	K00650.01	CAND	0.949	0.999	0.858	0.811	0

9.5 Comparison of predictions from Shallue and Vanderburg (2018) to LR/DT predictions

The last machine learning paper on kepler data Shallue and Vanderburg (2018) published a table (4) captioned "Summary of new Kepler TCEs predicted to be planets by our model with probability greater than 0.8.". Table 14 compares these results against the results obtained in section 9.3.

Shallue and Vanderburg (2018) declared the first and fourth models to be confirmed exoplanets 11442793,4852528 and the model 8480285 a strong candidate.

Six of the top nine ⁹ results recovered by Shallue and Vanderburg (2018) have also been recovered by our model and the results are roughly comparable which reinforces the confidence in the predictions made here. However the model in this paper has more predictions as it looked at a much larger number of systems.

There was not time (or space in technical report) to examine each planet individually but this was foreseen as future work.

Table 14: Comparison of 'new' TCE's predicted by Shallue vs TCE's predicted by LR and DT

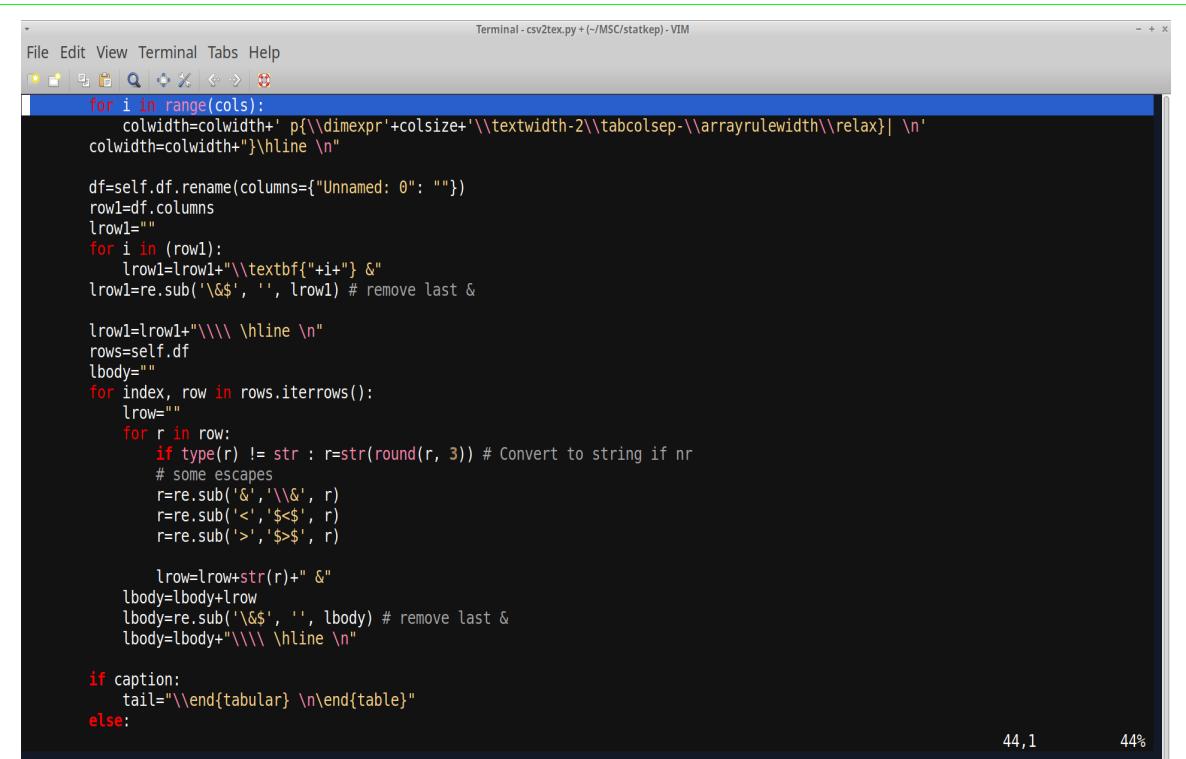
Kepid	Prediction Shallue and Vanderburg (2018)	Prediction (LR)	Prediction (DT)	koi_disp
11442793	0.942	None	None	-
8480285	0.941	0.971	0.948	CAND
11568987	0.92	0.805	0.741	CAND
4852528	0.916	None	None	-
5972334	0.896	0.915	0.933	CAND
10337517	0.86	0.697	0.88	CAND
11030475	0.858	0.907	0.875	CAND
4548011	0.852	0.575	0.601	CAND
3832474	0.847	None	None	-

⁹The missing three cases may be due to the lower S/N noise threshold used by Shallue and Vanderburg (2018)

10 Helper Programs

The programs Treat4.py and Treat5.py are used to present results and rely on several helper classes shown below.

- csv2tex.py shown in Figure 29 converts a csv file to a latex table and can be imported as a class or called independently.
- rpt2tex.py shown in Figure 30 creates a latex report using files produced by matplotlib.
- myfit.py shown in Figure 31 fits the or overfits the models and saves to a pickle file.



A screenshot of a terminal window titled "Terminal - csv2tex.py +(-/MSC/statkep) - VIM". The window contains the Python code for the `csv2tex.py` script. The code uses the `pandas` library to read a CSV file and convert its data into a LaTeX tabular environment. It handles column widths, row spans, and various data types (integers, floats, strings) by applying appropriate LaTeX escaping and conversion rules. The code is well-structured with comments explaining the logic for handling different data types and creating the final LaTeX output. The terminal window also shows the status bar at the bottom indicating the current line (44,1) and percentage (44%).

```
for i in range(cols):
    colwidth=colwidth+' p{\dimexpr'+colsizes+'\\textwidth-2\\tabcolsep-\\arrayrulewidth\\relax} | \n'
    colwidth=colwidth+"}\\hline \n"
df=self.df.rename(columns={"Unnamed: 0": ""})
rows=df.columns
lrow1=""
for i in (rows):
    lrow1=lrow1+"\\textbf{"+"i+"} &
    lrow1=re.sub('\\$', ' ', lrow1) # remove last &

    lrow1=lrow1+"\\\\ \\hline \n"
rows=self.df
lbody=""
for index, row in rows.iterrows():
    lrow=""
    for r in row:
        if type(r) != str : r=str(round(r, 3)) # Convert to string if nr
        # some escapes
        r=re.sub('&', '\\&', r)
        r=re.sub('<', '$<$', r)
        r=re.sub('>', '$>$', r)

        lrow=lrow+str(r)+" &
    lbody=lbody+lrow
    lbody=re.sub('\\$', ' ', lbody) # remove last &
    lbody=lbody+"\\\\ \\hline \n"

if caption:
    tail="\end{tabular} \n\end{table}"
else:
```

Figure 29: csv2tex.py: Converts a csv file to a latex table and can be imported as a class or called independently

```

File Edit View Terminal Tabs Help
Terminal - rpt2tex.py (~/MSC/statkep) - VIM
def ROCs(self,fpic1,fpic2,fpic3,fpic4,fcsv,caption):
    csvfile=csv2tex.csv2tex(fcsv)
    rpttab=csvfile.generatetab()
    csvoutput=csvfile.csvtex()
    csvoutput=csvfile.create_tex(caption="")
    label=self.label
    ftex=label+".tex"

    output="\n\\begin{figure}[H]\n"
    "\\begin{mdframed}[linecolor=green]\n"
    "\\centering\n"
    "\\begin{subfigure}{.49\\textwidth}\n"
    "\\includegraphics[width = 1\\textwidth]{%s}\n"
    "\\end{subfigure}\n"
    "\\begin{subfigure}{.49\\textwidth}\n"
    "\\caption{ %s }\n"
    "\\label{fig:%s}\n"
    "\\end{mdframed}\n"
    "\\end{figure}" %(fpic1,fpic2,fpic3,fpic4,rpttab,caption,label)
    output=csvoutput+output

```

65,1 19%

Figure 30: rpt2tex.py: Creates a latex file using files produced by matplotlib see Figure 19

```

File Edit View Terminal Tabs Help
Terminal - admin@martin-UX305FA: ~/MSC/statkep
Methods defined here:

__init__(self, fit, model='RF')
    At initialization the user select the input file and the model.
    If the input file begins with 'data/bT' so must the model name
    A name for the output files is generated based on these names.

fit_all(self)
    Run the model against all data / do split in train/test

fit_model(self, fit)
    Train the model and pickle results.

overfit_results(self)
    Run the model against 90% X_train,y_train data
    The resulting dataframe should be similar to predict_results
    If it is not (too good) this indicates overfitting

predict_results(self)
    Run the model against 10% X_test,y_test data

train_split(self)

-----
Data descriptors defined here:

__dict__
    dictionary for instance variables (if defined)

__weakref__

```

Figure 31: myfit.py: Generates models and save in pickle

11 Conclusion

This manual contains data which could not be included in the original report due to lack of space, including results. The appendix shows original R code which was not used and a description of the DV used for training (koi_disposition).

12 Appendix

12.1 Initial comparisons using R caret

Initially evaluation was performed using the R caret package. Naïve Bayes and Random Forest were compared. The code was developed in rstudio as shown in Figure 32. R proved to be less consistent than python and was abandoned by it was useful for initial insights and is included for completeness.

12.1.1 Naïve Bayes

Naïve Bayes is not ideal for datasets with many numeric features Lantz (2019).

12.1.2 Random Forest

The R package caret was tuned for the optimum values of mtry and ntree using uncleaned data 5. An accuracy of 0.913 was achieved using 1000 ntree and 6 retries see Table 15

Table 15: Initial Random Forest results after merging of TCE and KOI tables

RF_trainer	ntree	mtry	Accuracy
data/0_0.3TCE2_RF_500_6.rds	500	6	0.911354581673307
data/0_0.3TCE2_RF_500_8.rds	800	8	0.909860557768924
data/0_0.3TCE2_RF_800_10.rds	800	10	0.911354581673307
data/0_0.3TCE2_RF_1000_6.rds	1000	6	0.913346613545817
data/0_0.3TCE2_RF_1000_8.rds	1000	8	0.911852589641434
data/0_0.3TCE2_RF_1000_10.rds	1000	10	0.909860557768924

12.1.3 Results of comparison between Naïve Bayes and Random Forest

Random Forest section 12.1.2 achieved an accuracy of 0.913 without data cleaning. Removing noise affected these results little as it did with python sklearn.

Naïve Bayes achieved an accuracy of 0.785 using uncleaned data. Multivariate noise was then reduced using Mahalanobis calculations with ibm-spss and Naïve Bayes accuracy increased to 0.832.

These results indicate tree methods such as random forest are robust to noise and produce better results while requiring less data cleaning.

RStudio interface showing R code for generating RF results:

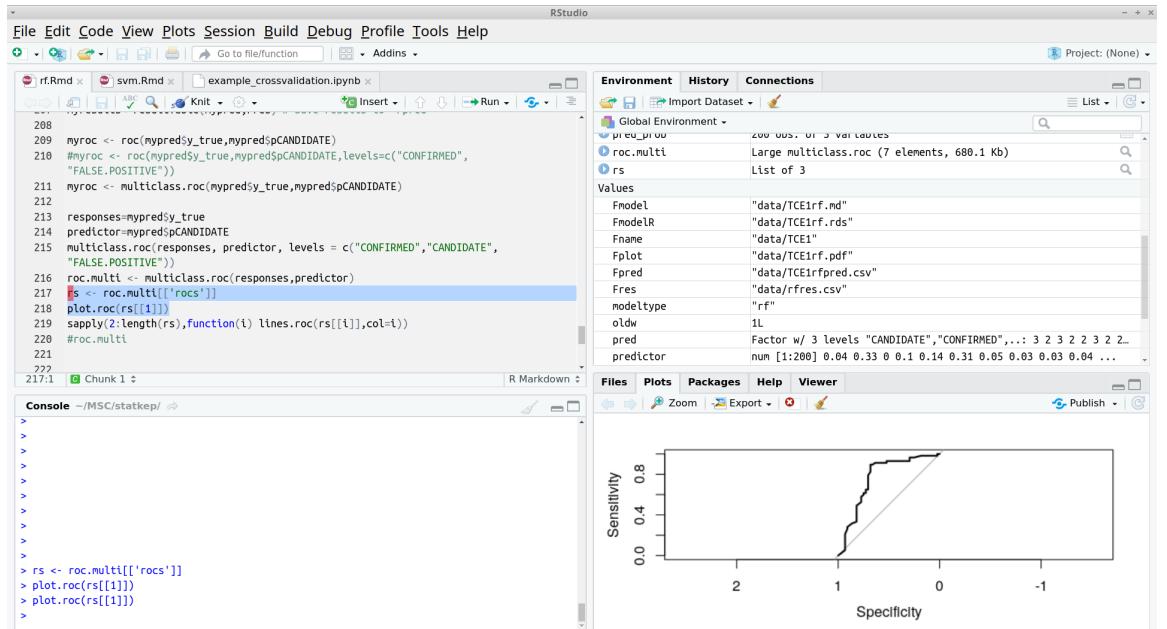
```

File Edit Code View Plots Session Build Debug Profile Tools Help
r.Rmd svm.Rmd example_crossvalidation.ipynb
48 resultable <- function(mypred,Fres) {
49 #   results<-caret::confusionMatrix(mypred$y_true,mypred$y_pred)
50 cmatrix <- caret::confusionMatrix(mypred$y_pred, mypred$y_true,positive='CONFIRMED')
51 
52 #   cmatrix<-confusionMatrix(mypred$y_true,mypred$y_pred)
53 overall<-round(as.matrix(cmatrix, what = "overall"),2)
54 classes<-round(as.matrix(cmatrix, what = "classes"),2)
55 # AUC
56 myroc <- roc(mypred$y_true,mypred$pCANDIDATE)
57 # Aucauc(myroc)
58 end_time <- Sys.time()
59 model_time<-as.numeric(round(difftime(end_time, start_time,units="secs"),3))
60 df<-data.frame(ifile<-c(FileName,
61 Accuracy<-c(overall["Accuracy",1]),
62 Accuracy<-(round(overall["Accuracy",1]),
63 Auc<-auc(mycroc),
64 Kappa<-c(overall["Kappa",1]),
65 F1<-c(classes["F1","CONFIRMED"]),
66 ntime_sec<-(model_time)
67 #Sensitivity <-c(classes["Sensitivity",1]),
68 )
69 
70 if(file.exists(Fres)){
71   write.table(format(df, digits=3),Fres, sep = ",",row.names=FALSE, col.names=
72 = FALSE, append = T)
73 } else{
74   write.csv(df, file = Fres, row.names=FALSE)
75 }
76 # Sort by Accuracy and remove duplicates
77 # df<-read.csv(Fres)
78 # df<-arrange(df,desc(Accuracy)) # sort by Accuracy
223:87 [r] Chunk 1: R Markdown

```

Environment pane shows variables and functions defined in the session.

(a) R code to generate RF results



(b) R code to create a simple ROC curve

Figure 32: R code on R studio to implement Random Forest

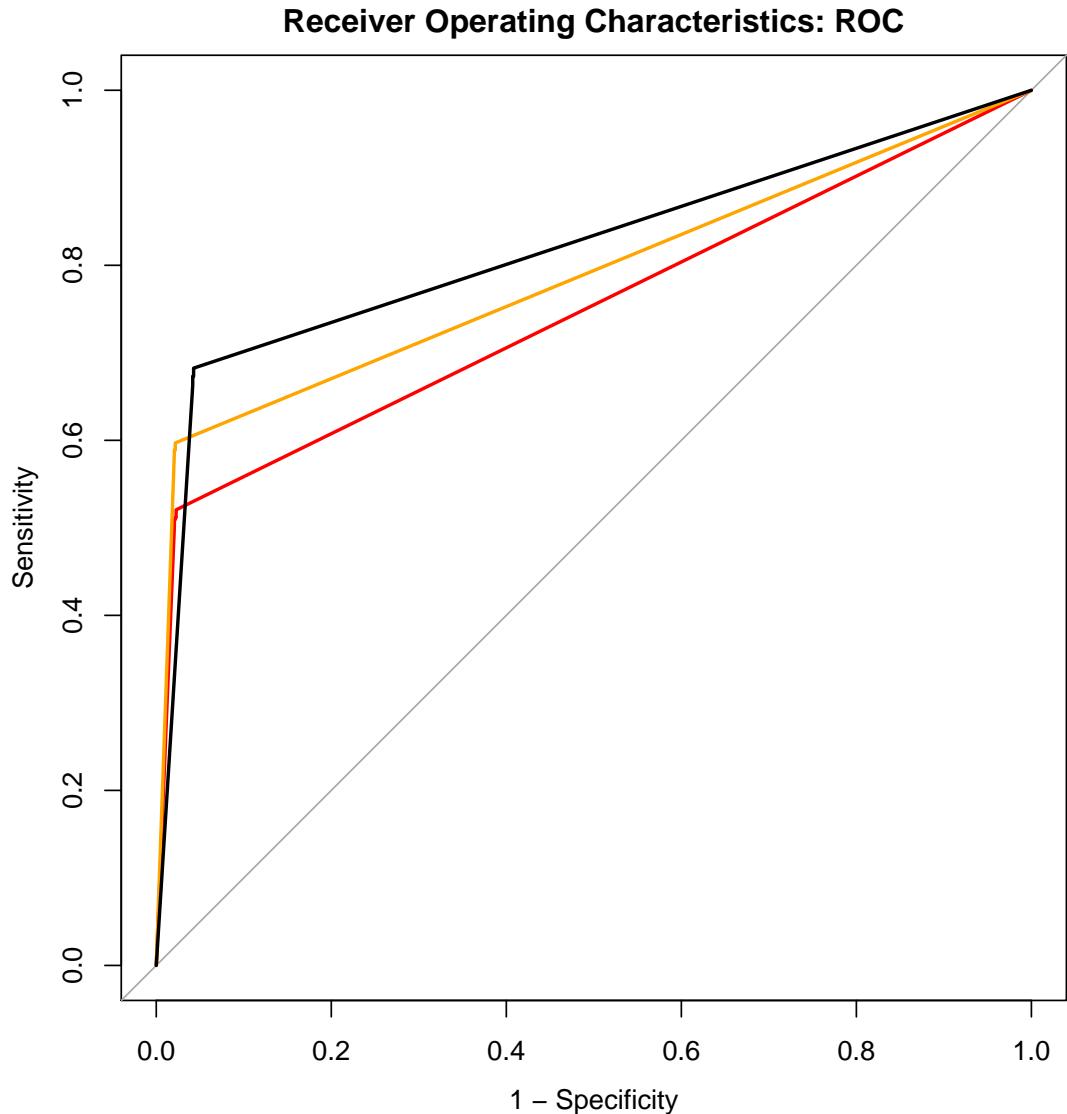


Figure 33: Exploratory Results: Initial comparison using Naïve Bayes
 TCE1mah1.csv -after removing outliers using Mahalanobis distance
TK.csv-before changing _err to _sn (signal to noise)

12.2 Data Description of DV (koi_disposition) and IVs

Table 16 shows the detailed description of the single DV from NASA ⁴ ⁵. An explanation of all IV's before modification are on the NASA website ², ³

Table 16: Detailed description of the DV koi_dispostion

Name	Description Kepler Object of Interest (TCE)	Details ⁵
koi_disposition	Exoplanet Archive Disposition	The category of this KOI from the Exoplanet Archive. Current values are CANDIDATE, FALSE POSITIVE, NOT DISPOSITIONED or CONFIRMED. All KOIs marked as CONFIRMED are also listed in the Exoplanet Archive Confirmed Planet table. Designations of CANDIDATE, FALSE POSITIVE, and NOT DISPOSITIONED are taken from the Disposition Using Kepler Data.

References

- Coughlin, J. L., Mullally, F., Thompson, S. E., Rowe, J. F., Burke, C. J., Latham, D. W., Batalha, N. M., Ofir, A., Quarles, B. L., Henze, C. E., Wolfgang, A., Caldwell, D. A., Bryson, S. T., Shporer, A., Catanzarite, J., Akeson, R., Barclay, T., Borucki, W. J., Boyajian, T. S., Campbell, J. R., Christiansen, J. L., Girouard, F. R., Haas, M. R., Howell, S. B., Huber, D., Jenkins, J. M., Li, J., Patil-Sabale, A., Quintana, E. V., Ramirez, S., Seader, S., Smith, J. C., Tenenbaum, P., Twicken, J. D. and Zamudio, K. A. (2016). Planetary Candidates Observed by Kepler. VII. The First Fully Uniform Catalog Based on the Entire 48-month Data Set (Q1-Q17 DR24), *The Astrophysical Journal Supplement Series* **224**: 12.
URL: <http://adsabs.harvard.edu/abs/2016ApJS..224...12C>
- Dziuban, C. D. and Shirkey, E. C. (1974). When is a correlation matrix appropriate for factor analysis? Some decision rules, *Psychological Bulletin* **81**(6): 358–361. Place: US Publisher: American Psychological Association.
URL: <https://psycnet.apa.org/record/1974-28961-001>
- Géron, A. (2017). *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*, O'Reilly Media, Sebastopol, CA.
- Lantz, B. (2019). *Machine Learning with R;Expert techniques for predictive modeling*, 3rd edn, Sage publications.
- Olson, R. S. and Moore, J. H. (2019). TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning, in F. Hutter, L. Kotthoff and J. Vanschoren (eds), *Automated Machine Learning: Methods, Systems, Challenges*, The Springer Series on

Challenges in Machine Learning, Springer International Publishing, Cham, pp. 151–160.

URL: https://doi.org/10.1007/978-3-030-05318-5_8

Randal S. Olson and Jason H. Moore (2018). TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning, *Automated Machine Learning : Methods, Systems, Challenges* p. 151. Place: Cham Publisher: Springer International Publishing.

Shallue, C. J. and Vanderburg, A. (2018). Identifying Exoplanets with Deep Learning: A Five-planet Resonant Chain around Kepler-80 and an Eighth Planet around Kepler-90, *The Astronomical Journal* **155**(2): 94. Publisher: American Astronomical Society.

URL: <https://doi.org/10.3847%2F1538-3881%2Fa9e09>